FACULTAD 4

Componentes gráficos para la representación de contadores eléctricos en el SCADA SAINUX

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

AUTOR: Gustavo González Díaz

TUTORES:

Ing. Rafael Alejandro Pérez Ordoñez

Ing. Carlos Silva Álvarez

La Habana, 2017


“El conocimiento y la tecnología es de especial relieve en nuestra agenda, porque en él abordamos los problemas que deciden, en buena medida, el futuro de nuestros países.”
Fidel Castro Ruz

Declaración de autoría

Declaro ser el único autor de este trabajo y autorizo hacer uso del mismo en su beneficio.

Para que así conste, firmo el presente a los ___ días del mes de _____ del año 2018.

Firma de autor: Gustavo González Díaz _____

_____ _____

Firma del tutor(a) Firma del tutor(a)

Datos de contacto

Autor:

Gustavo González Díaz

Universidad de las Ciencias Informáticas, La Habana, Cuba.

Email: ggonzalez@estudiantes.uci.cu

Síntesis de los tutores:

Tutor: Ing. Rafael Alejandro Pérez Ordoñez.

Síntesis: Graduado de la Universidad de las Ciencias Informáticas, tres años de experiencia en el desarrollo de sistemas SCADA.

Correo electrónico: rafaelalejandro@uci.cu

Tutor: Ing. Carlos Silva Álvarez.

Síntesis: Graduado de la Universidad de las Ciencias Informáticas.

Correo electrónico: silva@uci.cu

Agradecimientos: A mi familia y amigos.

Dedicatoria: A todas esas personas que me dieron su apoyo incondicional.

Resumen:

En el Centro de Informática Industrial (CEDIN) de la Universidad de las Ciencias Informáticas se desarrolla el SCADA SAINUX, el cual está compuesto por varios módulos entre los cuales se encuentra el módulo Interfaz Hombre Maquina. Actualmente el sistema SCADA SAINUX puede ser desplegado en varios sectores de la industria como el sector petrolero y el eléctrico. Para la representación de componentes eléctricos en el diseño de despliegues este sistema se apoya de figuras simples como líneas y curvas haciendo engorroso el trabajo con los mismos.

El siguiente trabajo tiene como objetivo desarrollar componentes eléctricos que permitan incrementar los grados de representación de los símbolos que conforman las redes eléctricas en la Interfaz Hombre Máquina del sistema SCADA SAINUX, con el uso de tecnologías y herramientas como: QtCreator, Visual Paradigm y lenguaje de programación C++. Como resultados se obtuvieron 16 componentes gráficos de tipo contador eléctrico que permiten representar los símbolos que conforman los circuitos eléctricos, lográndose una solución con mayores prestaciones y calidad en la representación de las redes eléctricas. Además, estos componentes disminuyen el tiempo de representación sobre el despliegue del sistema SCADA SAINUX.

Índice

Contenido

Índice de tabla

Índice de figuras

Introducción

El avance actual de las Tecnologías de la Información y las Comunicaciones (TIC) ha contribuido al desarrollo y crecimiento de las diferentes ramas de la industria. La automatización de los procesos industriales ha permitido sustituir el trabajo manual, disminuir el margen de error provocado por el factor humano y por tanto agilizar su desarrollo.

En los inicios de la automatización industrial los sistemas utilizados eran sencillos por esto a su vez con el paso del tiempo han evolucionado de forma exponencial. Hoy en día los procesos industriales se supervisan mediante Sistemas de Supervisión, Control y Adquisición de Datos (SCADA, por sus siglas en inglés).

Un sistema SCADA es una aplicación o conjunto de aplicaciones de software especialmente diseñadas para funcionar sobre ordenadores de control de producción, con acceso a la planta mediante la comunicación digital con instrumentos, actuadores e interfaz gráfica de alto nivel para el operador (pantallas táctiles, ratones o cursores, lápices ópticos, etc.) (1).

Cuba desarrolla un proceso de informatización de la sociedad con el objetivo de lograr el avance económico social de la nación. Un factor fundamental para dicho logro es el surgimiento de la Universidad de las Ciencias Informáticas (UCI), la cual cuenta con centros de desarrollo que proveen la exportación de software.

En el Centro de Informática Industrial (CEDIN) de dicha universidad se desarrolla el sistema de Supervisión, Control y Adquisición de Datos: SCADA SAINUX. Este sistema permite el monitoreo y el control de un área de trabajo a grandes distancias. La instalación del sistema SCADA SAINUX necesita un hardware de señal de entrada y salida, sensores y actuadores, controladores, redes, comunicaciones y base de datos, este cuenta con varios módulos como son Seguridad, BDH, Adquisición, Configuración y HMI.

El módulo HMI en el SCADA SAINUX 2.0 se encarga de representar, en un ordenador, los procesos que ocurren en el campo; muestran los componentes implicados, los sensores, las estaciones remotas, y el sistema de comunicación dándole al operador total control. Por su parte, permite al operador estar en contacto directo con el sistema, realizar la supervisión y el control del proceso en general. Trabaja sobre dos entornos: el entorno de configuración (EC), donde los mantenedores realizan la configuración de la información específica del área que se desea supervisar y diseñan los despliegues, los cuales hacen uso de los componentes gráficos que permiten simular los procesos de campo; y por otro lado, el entorno de visualización (EV), donde el operador puede supervisar y controlar la configuración realizada en el EC, interactuando con los componentes gráficos para emitir control sobre el sistema, monitorear los cambios de estado de las variables, gestionar alarmas y generar reportes.

Tanto para el EC como para el EV, el HMI provee una biblioteca de componentes gráficos que permiten recrear de una manera lo más real posible los procesos de campo. Estos gráficos pueden ser simples figuras geométricas como: línea, rectángulo, elipse, texto, polígono, polilínea, curvas; pero también muy complejas como los componentes de hardware utilizados en la industria que se desean supervisar como son: válvulas, generadores, motores de combustión, entre otros.

Entre los sinópticos gráficos usados en la representación de circuitos eléctricos existen los contadores eléctricos, los cuales no se encuentran en la biblioteca de componentes gráficos del HMI SAINUX 2.0. Para solventar dicha situación el mantenedor utiliza figuras básicas o imágenes para la representación de estos. Esto ocasiona varias desventajas como son:

- El proceso de configuración del circuito eléctrico se hace más lento y complicado mediante el uso de figuras básicas para la representación gráfica.
- Mediante el empleo y agrupación de figuras básicas para la representación de los contadores eléctricos, el componente gráfico resultante no puede ser guardado como un nuevo componente, afectándose su reutilización en otros despliegues.
- La agrupación de figuras básicas no puede ser tratada como un solo componente afectando la personalización de este.

Teniendo en cuenta la situación problemática planteada anteriormente, se define como **problema a resolver**: ¿Cómo contribuir a la representación de componentes gráficos de contadores eléctricos en el HMI del SCADA SAINUX 2.0?

Con el fin de resolver este problema se define como **objetivo general**: Desarrollar una solución integrada al SCADA SAINUX 2.0 que permita la representación de contadores eléctricos en el módulo de HMI.

El **objeto de estudio** de la presente investigación lo constituye: El proceso de visualización y configuración de componentes gráficos para procesos industriales.

El **campo de acción** está enmarcado en: Componentes gráficos para la representación de símbolos de instrumentación eléctrica.

Para dar cumplimiento al objetivo general se definen las siguientes tareas de investigación:

- Elaboración del marco teórico conceptual relacionado con los aspectos teóricos que sustentan la investigación.
- Realización del análisis y diseño de la propuesta de solución.

- Desarrollo de la propuesta de solución como soporte al modelo.
- Ejecución de pruebas funcionales a la implementación.
- Validación de la implementación a través de los métodos definidos en la investigación.

Para la realización de estas tareas se emplearon los métodos de investigación que se describen a continuación:

Métodos teóricos:

- **Analítico-sintético:** Para el estudio de los conceptos empleados en los sistemas SCADA, analizando todos los documentos elaborados, para la extracción de los elementos más importantes.
- **Histórico-lógico:** Para la comprensión de los antecedentes y las tendencias actuales referidas a la evolución en el mundo de los sistemas SCADA.

Métodos empíricos:

- **Observación:** Se puso en práctica este método, para conocer el funcionamiento existente en los despliegues del proyecto Continuidad del Sistema de Automatización Industrial UX (SAINUX 2.0), mediante el comportamiento de los dispositivos de campo en las propiedades de los objetos gráficos y sumarios empleados para la toma de decisiones de los operadores.
- **Consulta bibliográfica:** Empleada para consultar las fuentes de información relacionadas con los tipos de los sumarios y objetos gráficos visualizados en los entornos de escritorio de los HMI en sistemas SCADAS.

El presente documento está estructurado en tres capítulos:

Capítulo 1. Fundamentación teórica: Se realiza un esbozo teórico centrado en los conceptos y características fundamentales de los componentes gráficos para el proyecto Continuidad del Sistema de Automatización Industrial UX (SAINUX 2.0) para la representación de símbolos de instrumentación eléctrica.

Capítulo 2. Análisis y diseño de la solución propuesta: Se analiza la solución propuesta, se realiza el levantamiento de requisitos funcionales y no funcionales apoyándose en la metodología de desarrollo escogida. Al mismo tiempo se muestra la definición de las historias de usuario que regirán el desarrollo de la solución propuesta, y se realiza la descripción del diseño.

Capítulo 3. Implementación y validación de la solución propuesta: En este capítulo se describe la fase de implementación del sistema, según la metodología propuesta. Se realiza la fase de pruebas, a la cual se somete el producto para validar su correcto funcionamiento y de esta manera permitir comprobar que el mismo cumple con todos los requerimientos.

Capítulo 1: Fundamentación teórica.

1.1 Introducción

En este capítulo se hace un estudio del estado del arte, partiendo de los conceptos asociados a un sistema SCADA. Se explican los módulos y el funcionamiento de un sistema SCADA para la gestión y control de datos, enfatizando en el módulo HMI.

1.2 Sistemas SCADA

SCADA por sus siglas en inglés "*Supervisory Control And Data Acquisition*", (sistemas para la Supervisión, Control y Adquisición de datos), son sistemas basados en computadoras que permiten supervisar y controlar datos a grandes distancias, donde el lazo de control es generalmente cerrado por el operador. Proporciona comunicación con los dispositivos de campo (controladores autónomos, autómatas programables, etc.) y controla el proceso de forma automática desde la pantalla del ordenador. Además, provee toda la información que se genera en el proceso productivo a diversos usuarios, tanto del mismo nivel como de otros supervisores dentro de la empresa: control de calidad, supervisión y mantenimiento (2).

SCADA es la tecnología que permite a los usuarios recolectar la información de una o más instalaciones distantes, además de enviar instrucciones de control a esas instalaciones. Un SCADA hace innecesaria la presencia física de los operadores en los sitios distantes. Incluye interfaces para operadores y permite el monitoreo de los datos relacionados con los procesos (3).

De forma general los sistemas SCADA permiten el monitoreo y control de los procesos industriales a través de un ordenador, así como el almacenamiento de los datos referentes a dichos procesos para su posterior análisis. En la actualidad tienen numerosas aplicaciones dentro de las que se pueden mencionar: el control de oleoductos, sistemas de transmisión de energía eléctrica, yacimientos de gas y petróleo, redes de distribución de gas natural, subterráneos, generación energética, sistemas de distribución de agua, entre otros.

Los sistemas SCADA ofrecen una serie de prestaciones que van desde el control y supervisión de procesos en el nivel de campo hasta brindar soporte a los niveles de gestión y administración. Entre ellas se puede encontrar las siguientes:

- **Monitorización:** realizan la representación de datos del proceso en tiempo real, es decir las variables que se leen de los dispositivos: temperatura, presión, etc.

- **Supervisión remota:** permite conocer el estado y desempeño del proceso desde estaciones centrales. Resulta muy útil en procesos distribuidos en amplias locaciones, y permite coordinar labores de control de calidad y mantenimiento.
- **Control remoto de instalaciones y equipos:** posibilita cambiar datos claves del proceso directamente desde el ordenador (abrir o cerrar válvulas, encender motores, etc). También permite ajustar los valores, parámetros y algoritmos de control.
- **Visualización dinámica:** genera imágenes dinámicas que representan de manera intuitiva el comportamiento del proceso, reflejando los elementos de la planta. En estos gráficos también se pueden encontrar curvas y tablas de los datos y estados del sistema en el tiempo.
- **Adquisición y registro histórico de datos:** los datos adquiridos son procesados en tiempo real y almacenados en bases de datos, de manera que puedan ser analizados posteriormente a fin de evaluar el desempeño del sistema, así como realizar el diagnóstico y prevención de fallas.
- **Representación de señales de alarma:** por medio de las señales de alarma el sistema informa al operador la presencia de una falla o condición indeseable en el proceso, estas señales pueden ser visuales y/o sonoras.
- **Programación de aplicaciones:** existe la capacidad de programar informes, estadísticas o recetas para los autómatas.
- **Comunicación entre aplicaciones:** permiten el intercambio de información con diversas aplicaciones (4).

Pueden funcionar de manera centralizada o distribuida, para ello se agrupan las funcionalidades en módulos que pueden ser instalados en distintos servidores o nodos físicos pero que mantienen estrecha comunicación. Entre estos módulos se pueden destacar:

- **Configuración:** permite al usuario definir el entorno de trabajo de su SCADA, adaptándolo a la aplicación particular que se desea desarrollar.
- **Interfaz gráfica del operador**: proporciona al operador las funciones de control y supervisión de la planta. El proceso se representa mediante despliegues y gráficos generados desde el editor incorporado en el SCADA o importados desde otra aplicación durante la configuración del paquete.
- **Módulo de procesamiento:** ejecuta las acciones de mando preprogramadas a partir de los valores actuales de variables leídas.
- **Gestión y archivo de datos:** se encarga del almacenamiento y procesado ordenado de los datos, de forma que otra aplicación o dispositivo pueda tener acceso a ellos.

- **Comunicaciones**: se encarga de la transferencia de información entre la planta y la arquitectura hardware que soporta el SCADA, y entre esta y el resto de elementos informáticos de gestión (5).

En la UCI se desarrolla el sistema SCADA SAINUX por el CEDIN para su despliegue en las distintas industrias nacionales, utilizando tecnología libre. Este sistema está integrado por funcionalidades y módulos que caracterizan a un sistema SCADA.

Módulos del sistema SCADA SAINUX:

- **Comunicación:** Este módulo representa la capa de software encargada de la comunicación entre los diferentes procesos distribuidos, de mediano y alto nivel.
- **Configuración:** Módulo encargado de almacenar, persistir y suministrar, la información base para el funcionamiento de los demás módulos.
- **Seguridad:** Permite a los usuarios autenticarse en el sistema, y de esta forma poder acceder sólo a los recursos que tiene asignado su rol.
- **Bases de Datos Históricos:** Es el que permite almacenar la sucesión de alarmas y eventos, así como la información recibida desde los dispositivos que se encuentran en el campo. Esta información puede ser consultada y presentada mediante gráficos de tendencia y reportes.
- **Adquisición o Procesamiento de Datos:** Es el responsable de recolectar y procesar datos desde los dispositivos de campo.
- **HMI:** Representa gráficamente los procesos con los que interactúa el usuario. A través de este módulo se puede visualizar el estado de las comunicaciones con los dispositivos de campo, los valores de las variables procesadas y las alarmas generadas.

1.2.1 Módulo Interfaz Hombre Máquina

La Interfaz Hombre-Máquina o HMI (*Human Machine Interface*) es el sistema que muestra los datos a un operador (humano) y a través de la cual este controla el proceso. Son interfaces gráficas que muestran información del proceso en tiempo real, utilizando diagramas esquemáticos, algunos contornos y hasta animaciones en pantalla o paneles. Los sistemas HMI facilitan mucho las tareas de diseño debido a que presentan herramientas para crear bases de datos dinámicas, permiten crear y animar pantallas de forma sencilla y además incluyen gran cantidad de librerías de objetos para representar los diferentes dispositivos de la industria como motores, tanques, indicadores, interruptores válvulas y bomba, signos de advertencias, señales de dirección, cámara de seguridad, entre otras (6).

1.2.2Tipos de ambiente.

- **Ambiente de configuración del HMI:** Permite definir el ambiente de trabajo del SCADA, adaptando mejor la aplicación al proceso que se desea supervisar. Es el ambiente donde el mantenedor configura la información específica del área que se desea supervisar, diseña los despliegues, los cuales haciendo uso de los componentes gráficos y las animaciones permiten simular los procesos de campo (7).
- **Ambiente de visualización del HMI:** El ambiente de ejecución se encarga de visualizar los componentes gráficos y las animaciones definidos en el ambiente de configuración. El ambiente de visualización es donde el operador puede supervisar y controlar la configuración realizada en el ambiente de configuración, comunicándose con los componentes gráficos para emitir control sobre el sistema y permitir monitorear los cambios de estado de las variables, la gestión de las alarmas y los reportes (7).

1.2.3 SCADAs eléctricos.

- **Citect SCADA:** Creado por Schneider Electric presenta una solución SCADA simplificada e intuitiva que redefine la experiencia operativa y brinda un contexto más rico para abordar la Gestión Situacional Anormal (ASM) en una variedad de aplicaciones industriales. Algunas de las características clave de Citect SCADA 2018 incluyen: Una extensa biblioteca de objetos configurables, un espacio de trabajo dedicado a la conciencia situacional, además de una carga de visualización y mejoras gráficas (8).
- **AggreGate SCADA / HMI:** Es un sistema SCADA de nueva generación para automatización industrial y de edificios, control de procesos, telemetría y diseño de interfaces hombre-máquina personalizadas. Los controladores para la mayoría de los protocolos estándar de la industria, como Modbus y OPC, permiten conectarse y comunicarse con controladores y dispositivos de miles de fabricantes diferentes (9).

A través de la investigación realizada se comprobó la magnitud que puede alcanzar un sistema de este tipo, cuantas variables se pudieran controlar simultáneamente y cuánta información puede llegar a generar el mismo. Por lo que se llega a la conclusión que los conceptos, flujos de acciones y componentes gráficos no puede ser reutilizado para la solución propuesta, ya que:

- Utiliza un marco de trabajo privativo.
- Su marco conceptual no puede ser utilizado libremente.
- El lenguaje de programación no es el utilizado en el SCADA SAINUX 2.0.
- Para su utilización, se debe comprar la licencia de uso.

- Es usado sobre sistemas operativos bajo licencias.

1.3 Componentes gráficos

El SCADA SAINUX cuenta con una paleta de componentes que mediante una correcta disposición de estos componentes sobre las secciones del diseño se puede definir la apariencia visual de los procesos que se deseen supervisar. Cada componente gráfico presenta propiedades las cuales definen su comportamiento sobre el sistema.

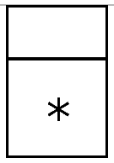
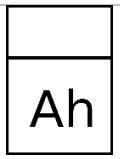
1.3.1 Contadores eléctricos.

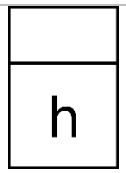
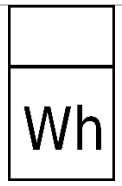
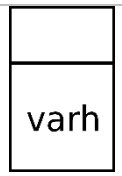
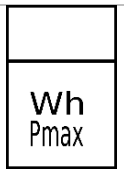
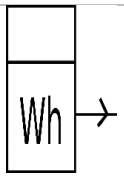
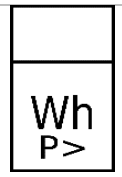
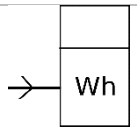
Los contadores eléctricos son dispositivos que miden el consumo de la energía eléctrica de un circuito o un servicio eléctrico. Se clasifican de la siguiente manera:

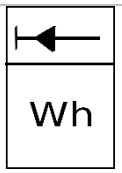
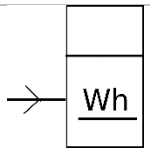
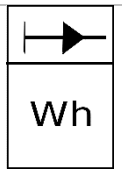
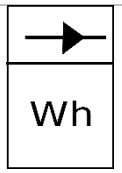
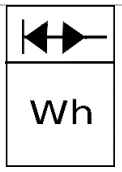
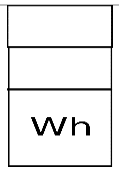
- **Contador de energía activa:** Este contador registra la cantidad de energía activa que las empresas suministradoras entregan al abonado. La unidad de medida es el kilovatio hora (kW/h.)
- **Contador de energía reactiva:** Si en la instalación del abonado hay receptores de carácter inductivo, se usa el contador de energía reactiva para calcular el factor de potencia medio de la instalación. La unidad de medida es kilovoltiamperio reactivo hora (10).

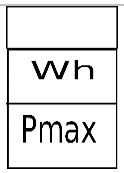
A modo de conclusión los contadores que se utilizan en la representación de los componentes eléctricos son todos contadores de energía activa menos el varihómetro que es un contador de energía reactiva. A continuación, se presenta una tabla con el nombre del contador, la figura que se utilizará para la representación del componente siguiendo las normas de la UNE (Unión Nacional Eléctrica), su descripción y unidad de medida:

Tabla 1 Descripción de los componentes.

Nombre	Figura	Descripción	Unidad de medida
Instrumento integrador		Se sustituye el asterisco por la letra o símbolo de la magnitud a contar.	-
Amperihorímetro		Aparato destinado a medir una cantidad de electricidad por integración de una intensidad en función del tiempo(11).	Ah (amperios-horas)

Contador horario		Mide el tiempo de funcionamiento de un equipo eléctrico en horas (12).	H (horas)
Varihorímetro		Instrumento destinado a medir la energía reactiva mediante la integración de la potencia reactiva con respecto al tiempo (13).	Wh (vatios-horas)
Varihómetro		Contador de energía reactiva y utiliza voltiamperios-reactivos-horas (13).	Varh (voltiamperios-reactivos-horas)
Contador de energía con indicador de máxima demanda		Medidor de energía equipado con un medio para indicar el valor promedio más alto de la potencia durante intervalos de tiempo sucesivos de igual duración (13).	Wh (vatios-horas)
Contador de energía con transmisor		Dispositivo que mide la potencia activa con la corriente que corre entre la entrada y la salida del equipo a medir (13).	Wh (vatios-horas)
Contador de exceso		Es un dispositivo que mide el consumo de energía eléctrica de un circuito o un servicio eléctrico (13).	Wh (vatios-horas)
Contador de energía con telemando		Es un dispositivo que mide el consumo de energía eléctrica de un circuito o un servicio eléctrico (13).	Wh (vatios-horas)

Contador de energía hacia barra ómnibus		Es un dispositivo que mide el consumo de energía eléctrica de un circuito o un servicio eléctrico que fluye hacia la barra ómnibus(Las barras ómnibus están concebidas para sus dispositivos de protección contra errores en el arco eléctrico) (13).	Wh (vatios-horas)
Contador de energía con telemando e impresión		Es un dispositivo que mide el consumo de energía eléctrica de un circuito o un servicio eléctrico (13).	Wh (vatios-horas)
Contador de energía desde barra ómnibus		Es un dispositivo que mide el consumo de energía eléctrica de un circuito o un servicio eléctrico que fluye hacia la barra ómnibus (13).	Wh (vatios-horas)
Contador de energía en una sola dirección		Es un dispositivo que mide el consumo de energía eléctrica para una sola dirección de un circuito o un servicio eléctrico (13).	Wh (vatios-horas)
Contador energía ambas direcciones		Es un dispositivo que mide el consumo de energía eléctrica para ambas direcciones de un circuito o un servicio eléctrico (13).	Wh (vatios-horas)
Contador energía tarifa doble		Es un dispositivo que mide el consumo de energía eléctrica de un circuito o un servicio eléctrico con la diferencia de tener 2 totalizadores (aparato que da el total de una serie de operaciones) en su parte visible (13).	Wh (vatios-horas)

Contador energía registrador máxima demanda		Es un dispositivo que mide el consumo de energía eléctrica de un circuito o un servicio eléctrico que tenga mayor demanda de energía eléctrica (13).	Wh (vatios-horas)

1.4 Normas para la representación de componentes eléctricos.

Las normas se definen según las necesidades de las regiones y sectores. Una norma está definida como regla o conjunto de reglas que hay que seguir para llevar a cabo una acción. Estas normas formalizan el uso de los símbolos gráficos y las reglas generales del sistema alfanumérico a seguir para identificar aparatos, diseñar esquemas de circuitos y montaje de instalaciones, dispositivos y aparatos eléctricos y electrónicos.

1.4.1 UNE-EN 60617 (IEC60617)

En los últimos años (1996 a 1999) se han visto modificados los símbolos gráficos para esquemas eléctricos, a nivel internacional con la norma IEC 60617, que se ha adoptado a nivel europeo en la norma EN 60617 y que finalmente se ha publicado en España como la norma UNE-EN 60617 (14). A continuación, se muestran las asociaciones que rigen los estándares internacionales utilizados en la documentación gráfica de los símbolos eléctricos.

- **DIN – Deutsches Institut für Normung:** Instituto alemán de normalización. Organismo nacional de normalización de Alemania con sede en Berlín. También se interpreta DIN con Deustches Industrie Norm Normalización para la industria alemana.
- **ANSI – American National Standards Institute:** Instituto Nacional Estadounidense de Estándares. Organización que supervisa el desarrollo de estándares para productos, procesos, servicios y sistemas en EEUU.
- **IEC – International Electrotechnical Commission:** Comisión Electrotécnica Internacional. Organización internacional de normalización en los campos eléctrico, electrónico y tecnologías afines (15).

La norma, está dividida las siguientes partes de la cual solo se utilizará la UNE-EN 60617-2 (Elementos de símbolos, símbolos distintivos y otros símbolos de aplicación general):

Parte	Descripción
UNE-EN 60617-2	Elementos de símbolos, símbolos distintivos y otros símbolos de aplicación general
UNE-EN 60617-3	Conductores y dispositivos de conexión
UNE-EN 60617-4	Componentes pasivos básicos
UNE-EN 60617-5	Semiconductores y tubos electrónicos
UNE-EN 60617-6	Producción, transformación y conversión de la energía eléctrica
UNE-EN 60617-7	Aparamenta y dispositivos de control y protección
UNE-EN 60617-8	Instrumentos de medida, lámparas y dispositivos de señalización
UNE-EN 60617-9	Telecomunicaciones : Conmutación y equipos periféricos
UNE-EN 60617-10	Telecomunicaciones : Transmisión
UNE-EN 60617-11	Esquemas y planos de instalación, arquitectónicos y topográficos.
UNE-EN 60617-12	Operadores lógicos binarios
UNE-EN 60617-13	Operadores analógicos

Figura 1: "Partes de la norma UNE-EN 60617 (IEC60617)(14)"

En la siguiente tabla se muestra como se definirán los colores de los componentes según el proceso por el que estará identificado cada componente eléctrico, estos colores son definidos por el Instituto Nacional Estadounidense de Estándares:

Tabla 2 Colores procesos en panel (16)

Color	**Significado genérico**	**Elementos asociados**
Negro	Emergencia	a) Paro.
		b) Alarma de muy alta prioridad.
		c) Cerrado.
		d) Desconectado (OFF).

Amarillo	Precaución	a) Condición anormal.
		b) Alarma de prioridad secundaria.
Verde	Seguridad	a) Operación normal.
		b) Arranque .
		c) Abierto.
		d) Conectado (ON).
Cyan (azul claro)	Estático significativo	a) Equipo de proceso en servicio.
		b) Tarjetas principales.
Azul	No esencial	a) Equipo de proceso en reserva.
		b) Tarjetas, tags, etc.
Magenta (púrpura)	Radiación	a) Alarmas de radiación.
		b) Valores dudosos.
Blanco	Datos dinámicos	a) Información de medida y estado .
		b) Mensajes del sistema .
		c) Tendencias.
		d) Paso secuencial activo.

1.5 Herramientas y tecnologías

1.5.1 Gráficos Vectoriales Escalables (SVG)

SVG son las siglas de *Scalable Vector Graphics*, que se podría traducir libremente al español como gráficos basados en vectores escalables. En otras palabras, es un formato gráfico basado en XML para crear archivos vectoriales en 2D, con un lenguaje de marcado por medio de etiquetas. Entre sus funcionalidades se tiene la capacidad de usar tres tipos de objetos gráficos: formas de vectores gráficos (entre las que se incluyen líneas, polígonos, poli línea, rectángulos, círculos o elipses), imágenes y texto. Además, a los objetos gráficos se le puede aplicar transformaciones (traslaciones, escala), animaciones y efectos de filtro.

Algunas ventajas de este formato son: no pierde calidad si se hace zoom, se puede escalar, se muestra de forma progresiva, no se tiene que esperar a que se descargue todo el archivo, los vectoriales por ser solamente parámetros matemáticos, suelen ser mucho menos pesados que una imagen rasterizada o de píxeles. Los gráficos vectoriales pueden ser transformados (estirar, rotar, mover, distorsionar) de una manera más sencilla y con menos requerimientos de memoria en el equipo (17).

Partiendo de las ventajas expuestas que brinda este formato de imagen se decide aplicarlo en el diseño de los componentes eléctricos en el SCADA SAINUX 2.0.

1.5.2 Inskcape

El uso de *Inkscape* en su versión 0.48 se debe a que su objetivo principal es crear una herramienta de dibujo potente y cómoda, totalmente compatible con los estándares XML, SVG y hojas de estilo en cascada (CSS, por sus siglas en inglés). Las características soportadas incluyen: formas, trazos, texto, marcadores, transformaciones y gradientes. *InkScape* es un editor de gráficos vectoriales de código abierto que usa el formato de archivo SVG. Este editor es una herramienta multiplataforma que se encuentra desarrollada principalmente para el sistema operativo GNU/Linux (18). La herramienta *Inkscape* proporcionará el desarrollo de los componentes eléctricos en formato SVG para su despliegue en el sistema SCADA.

1.5.3 Marco de Trabajo

Para la representación de los contadores eléctricos se hace necesario la utilización de un marco de trabajo (*framework*, en inglés), el cual se utiliza de base para la organización y desarrollo del software. Un *framework* puede incluir soporte de programas, bibliotecas, lenguaje interpretado, entre otras herramientas, para así ayudar a desarrollar los diferentes componentes de un proyecto.

El *framework* utilizado es Qt en su versión 4.8.2, el cual es multiplataforma para el desarrollo de aplicaciones. Estas pueden ser con o sin interfaz gráfica (19). Para el desarrollo del presente trabajo se hace uso de este

framework debido a que utiliza el lenguaje de programación C++ de forma nativa y es precisa para realizar todas las interfaces gráficas necesarias.

Qt incluye un amplio conjunto de componentes gráficos que proporcionan las funcionalidades estándar de interfaz gráfica de usuario, introduce una innovadora alternativa para la comunicación entre objetos, llamados señales y ranuras. Puede soportar las funcionalidades de interfaz de usuario que requieren las aplicaciones modernas, tales como menús, menús contextuales, arrastrar, soltar, y barras de herramientas. Propone el patrón de diseño modelo/vista para la representación gráfica de los datos, con la separación de las funcionalidades introducidas por esta arquitectura. El marco de trabajo Qt ofrece a los desarrolladores una mayor flexibilidad para personalizar la presentación de elementos y proporciona una interfaz de modelo estándar que permite una amplia gama de fuentes de datos.

1.5.4 Entorno de Desarrollo Integrado (IDE)

Para el desarrollo de este producto se requiere de un entorno de desarrollo integrado (IDE, por sus siglas en inglés), el cual puede denominarse como un entorno de programación que consiste en un editor de código y un compilador.

El IDE seleccionado para el desarrollo de la aplicación es Qt Creator en su versión 2.5.0. Qt Creator es un IDE multiplataforma para el desarrollo de aplicaciones que pueden o no tener interfaz gráfica. Este se centra en proporcionar características que ayudan a los nuevos usuarios del IDE a aprender y comenzar a desarrollar rápidamente (20).

Existen otras características importantes que presenta este IDE como la disponibilidad de código fuente, la excelente documentación organizada que provee en el QtAssistant y un editor para el diseño de formularios denominado QtDesigner (20).

Qt Creator cuenta con:

- Un editor de código con soporte para C++.
- Herramientas para la rápida navegación por el código.
- Resaltado de sintaxis y auto-completado de código.
- Soporte para refactorización de código.
- Paréntesis coincidentes y modos de selección.

1 .5.5 Metodología de desarrollo de software: Metodología AUP-UCI

El Proceso Unificado Ágil (AUP, por sus siglas en inglés) es una versión simplificada del Proceso Racional Unificado (RUP, por sus siglas en inglés). Este describe de una manera simple y fácil de entender la forma de desarrollar aplicaciones de software de negocio usando técnicas ágiles y conceptos que aún se mantienen válidos en RUP. La variación de la metodología AUP, denominada AUP-UCI se adapta al ciclo de vida definido para la actividad productiva de la UCI y es utilizada por el CEDIN en sus productos de software. El uso de esta técnica de modelado ágil permite encapsular los requisitos funcionales en Historias de Usuario (HU) o descripción de requisitos por procesos (21).

Fases de AUP-UCI:

Para el desarrollo del trabajo la metodología AUP-UCI cuenta con tres fases fundamentales que se exponen a continuación:

Tabla 3 Fases AUP-UCI

Fases	Objetivos
Inicio	✓ Planeación del proyecto. ✓ Estudio inicial de la organización cliente. ✓ Alcance del proyecto. ✓ Estimaciones de tiempo, esfuerzo y costo.
Ejecución	✓ Ejecución de las actividades requeridas para desarrollar el software. ✓ Ajuste de los planes del proyecto. ✓ Modelado del negocio. ✓ Obtención de requisitos. ✓ Se elaboran la arquitectura y el diseño. ✓ Se implementa y se libera el producto.

Cierre	✓ Resultados obtenidos. ✓ Cierre del proyecto.

Disciplinas de AUP-UCI

Tabla 4 Disciplinas de AUP-UCI

Disciplinas	Objetivos
Modelado de negocio (opcional)	Comprender los procesos de negocio.
Requisitos	Administración y gestión de los requisitos funcionales y no funcionales.
Análisis y diseño	Modelar el sistema.
Implementación	Construcción del sistema.
Pruebas internas	Verificar el resultado de la implementación.
Pruebas de liberación	Diseñar y ejecutar pruebas por una entidad externa certificadora de la calidad.
Pruebas de aceptación	Verificar que el software está listo.
Despliegue (Opcional)	Instalación, configuración, adecuación, y puesta en marcha.
Gestión y soporte	Conjunto de operaciones que se realizan para dirigir y administrar una empresa con el objetivo de sostener o mantener el negocio.

1 .5.6 Lenguaje de programación

Todo producto informático necesita un lenguaje de programación para su desarrollo, el cual se comunica con la computadora y orienta a la misma para que realice una tarea específica.

El lenguaje de programación seleccionado es C++ versión 98, con el objetivo de lograr una mayor compatibilidad con el SCADA SAINUX pues es el lenguaje en el que está desarrollado. C++ está considerado por muchos como un lenguaje de programación potente, debido a que permite el trabajo tanto a alto como a bajo nivel, logrando gran eficiencia en tiempo de ejecución y bajo consumo de memoria en los programas desarrollados, algo que requieren la mayoría de las aplicaciones (22).

1 .5.7 Visual Paradigm

La herramienta seleccionada para realizar el modelado de la solución es el Visual Paradigm en su versión 8.0. Visual Paradigm es una herramienta de Ingeniería de Software Asistida por Computación (CASE, por sus siglas en inglés). La misma propicia un conjunto de funcionalidades para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación (23).

1.6 Conclusiones parciales

- La elaboración del marco teórico brindó un acercamiento a los principales conceptos, técnicas y métodos a utilizar durante la investigación. Permitió profundizar los aspectos teóricos de sistemas SCADA, sus interfaces hombre-máquina y el análisis de los componentes que el mismo posee, dando como resultado de la investigación los colores que deben usar los componentes y la norma para formalizar la representación de los mismos.
- El estudio de la metodología y las herramientas a utilizar consolidó la propuesta tecnológica necesaria para la realización de los componentes.

Capítulo 2: Análisis y Diseño de la Solución.

2.1 Introducción

El presente capítulo tiene como objetivo describir las actividades realizadas durante el proceso de análisis y diseño, con el objetivo de modelar y dar respuesta al problema planteado. Se especifican los requisitos funcionales y no funcionales que debe cumplir el sistema para satisfacer las necesidades del cliente. Al mismo tiempo se muestra la definición de las historias de usuario que regirán el desarrollo de la solución propuesta, y se realiza la descripción del diseño.

2.2 Modelo del dominio

Un Modelo de dominio es un artefacto de la disciplina de análisis, construido con las reglas de UML durante la fase de concepción, en la tarea construcción del modelo de dominio, presentado como uno o más diagramas de clases y que contiene, no conceptos propios de un sistema de software sino de la propia realidad física. Los modelos de dominio pueden utilizarse para capturar y expresar el entendimiento ganado en un área bajo análisis como paso previo al diseño de un sistema, ya sea de software o de otro tipo. Similares a los mapas mentales utilizados en el aprendizaje, el modelo de dominio es utilizado por el analista como un medio para comprender el sector industrial o de negocios al cual el sistema va a servir (24).

El modelo de dominio es una representación de los conceptos clave de un área o problema. Se utilizará para la representación de los aspectos del mundo real o físico. A continuación, se muestra el modelo de dominio utilizado para la solución propuesta.

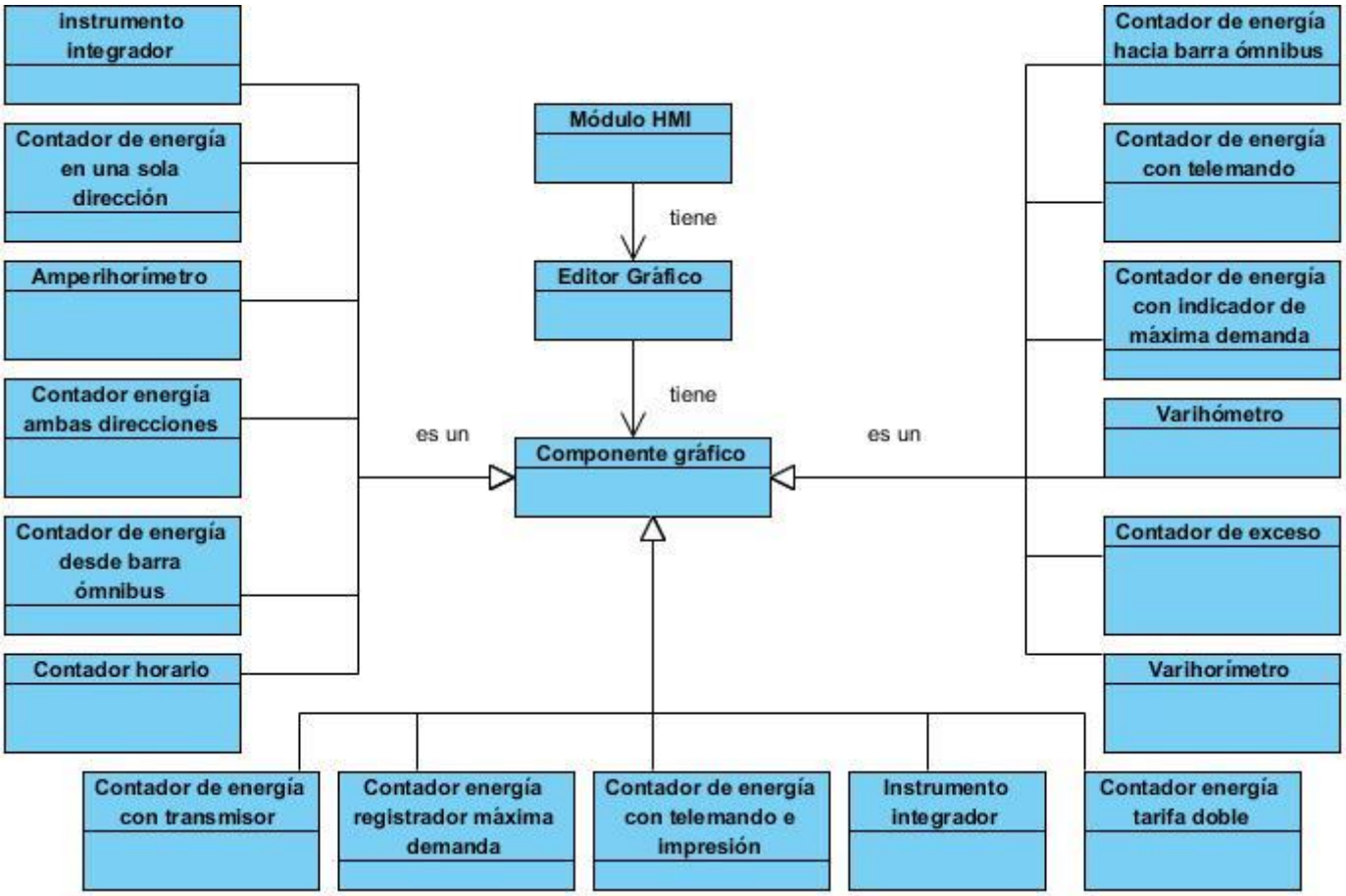


Figura 2: Modelo del dominio

Descripcion del modelo de dominio:

- **Módulo HMI:** Interfaz encargada de mostrar en un ordenador los procesos que ocurren en el campo.
- **Componente gráfico:** Representa de manera visual una serie de datos por medio de figuras o signos.
- **Editor gráfico:** interfaz donde se muestra la paleta de componentes disponibles para la configuración.
- **Instrumento integrador:** Clase interfaz de tipo instrumento integrador.
- **Amperihorímetro:** Clase de tipo amperihorímetro.
- **Contador horario:** Clase de tipo Contador horario.
- **Varihorímetro:** Clase de tipo varihorímetro.
- **Varihómetro:** Clase de tipo varihómetro.
- **Contador de energía con indicador de máxima demanda:** Clase de tipo contador de energía con indicador de máxima demanda.

- **Contador de energía con transmisor:** Clase de tipo contador de energia con transmisor.
- **Contador de exceso:** Clase de tipo contador de exceso.
- **Contador de energía con telemando:** Clase de tipo contador de energía con telemando.
- **Contador de energía hacia barra omnibus:** Clase de tipo contador de energía hacia barra ómnibus.
- **Contador de energía con telemando e impresión:** Clase de tipo contador de energía con telemando e impresión.
- **Contador de energía desde barra ómnibus:** Clase de tipo contador de energía desde la barra de ómnibus.
- **Contador de energía en una sola dirección:** Clase de tipo contador de energía en una sola dirección.
- **Contador energía ambas direcciones:** Clase de tipo contador energía ambas direcciones.
- **Contador energía tarifa doble:** Clase de tipo contador de energía con tarifa doble.
- **Contador energía registrador máxima demanda:** Clase de tipo contador de energia con registrador de máxima demanda.

2.3 Captura de requisitos.

Un requisito es una condición o capacidad que necesita el usuario para resolver un problema o conseguir un objetivo determinado. También se aplica a las condiciones que debe cumplir o poseer un sistema o uno de sus componentes para satisfacer un contrato, una norma o una especificación (25).

2.3.1 Requisitos funcionales (RF).

Los requisitos funcionales son una definición de los servicios que el sistema debe proporcionar, cómo debe reaccionar a una entrada particular y cómo se debe comportar antes situaciones particulares (25).

RF1: El sistema debe permitir visualizar el componente gráfico contador eléctrico en la paleta de componente del HMI.

RF2: El sistema debe permitir arrastrar el componente gráfico de tipo contador eléctrico hacia el despliegue.

RF3: El sistema debe permitir rotar el componente gráfico de tipo contador eléctrico en el despliegue.

RF4: El sistema debe permitir redimensionar el componente gráfico de tipo contador eléctrico en el despliegue.

RF5: El sistema debe mostrar todas las propiedades del componente gráfico de tipo contador eléctrico.

RF6: El sistema debe permitir actualizar las propiedades del componente gráfico de tipo contador eléctrico.

RF7: El sistema debe permitir mostrar el componente gráfico en el visualizador.

RF8: El sistema debe permitir eliminar el componente contador eléctrico del despliegue.

RF8: El sistema debe permitir configurar el color del componente contador eléctrico sobre el despliegue.

2.3.2 Requisitos no funcionales (RNF)

Requisitos del software:

- El software debe ser compatible con en el Sistema Operativo GNU-Linux en la distribución Debian en su versión 8.

Restricciones en el diseño y la implementación:

- Se debe hacer uso del *framework* Qt, en su versión 4.8.2.
- Como lenguaje de programación debe ser utilizado el lenguaje de programación C++.

2.3.3 Historias de usuario

Las Historias de Usuario son un enfoque de requerimientos ágil que se focaliza en establecer conversaciones acerca de las necesidades de los clientes. Son descripciones cortas y simples de las funcionalidades del sistema, narradas desde la perspectiva de la persona que desea dicha funcionalidad (26).

Tabla 5 Historia de usuario #1

Historia de Usuario	
Número: 1	**Nombre del requisito:** Visualizar el componente en la paleta de componentes del HMI.
Programador: Gustavo González Díaz.	**Iteración Asignada:** 2.
Prioridad: Alta.	**Tiempo Estimado:**1 semana.
Riesgo en Desarrollo: Problemas eléctricos, problemas técnicos.	**Tiempo Real:** 1 semana.

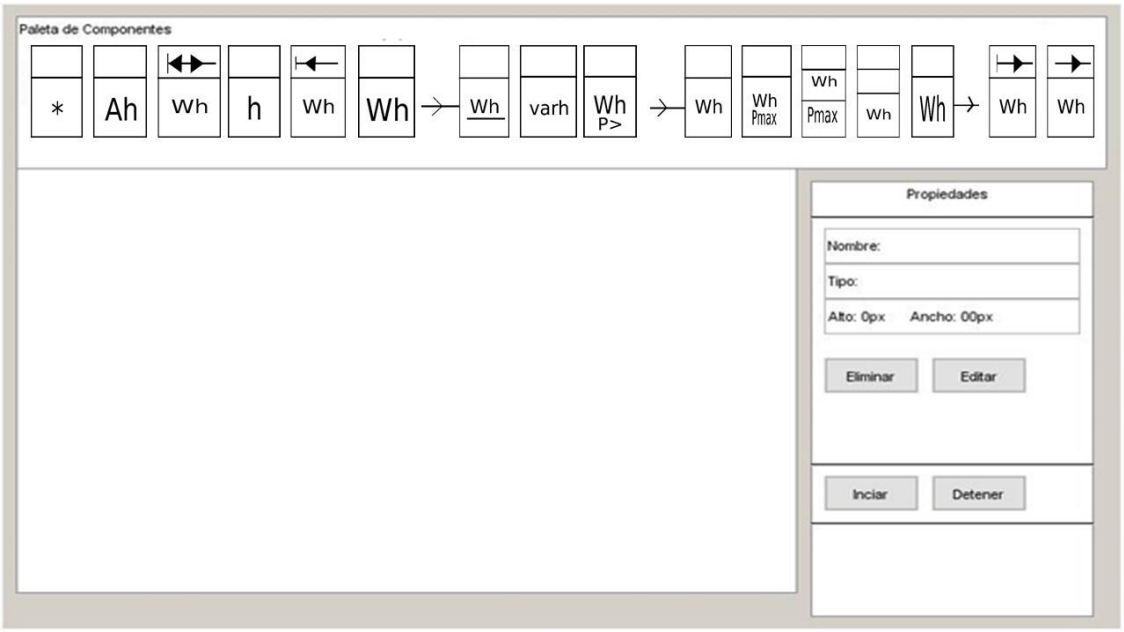
Descripción: El operador podrá visualizar los componentes de tipo contador eléctrico en la paleta de componentes del HMI.
Prototipo de interfaz: 

Tabla 6 Historia de usuario #2

Historia de Usuario	
Número: 2	**Nombre del requisito:** Arrastrar el componente hacia el despliegue.
Programador: Gustavo González Díaz.	**Iteración Asignada:** 1.
Prioridad: Alta.	**Tiempo Estimado:**1 semana.
Riesgo en Desarrollo: Problemas eléctricos, problemas técnicos.	**Tiempo Real:** 1 semana.
Descripción: El operador podrá adicionar el componente gráfico tipo contador eléctrico al despliegue del editor gráfico. Para la operación el operador debe hacer clic izquierdo sobre el componente deseado y arrastrarlo hacia el despliegue.	
Prototipo de interfaz:	

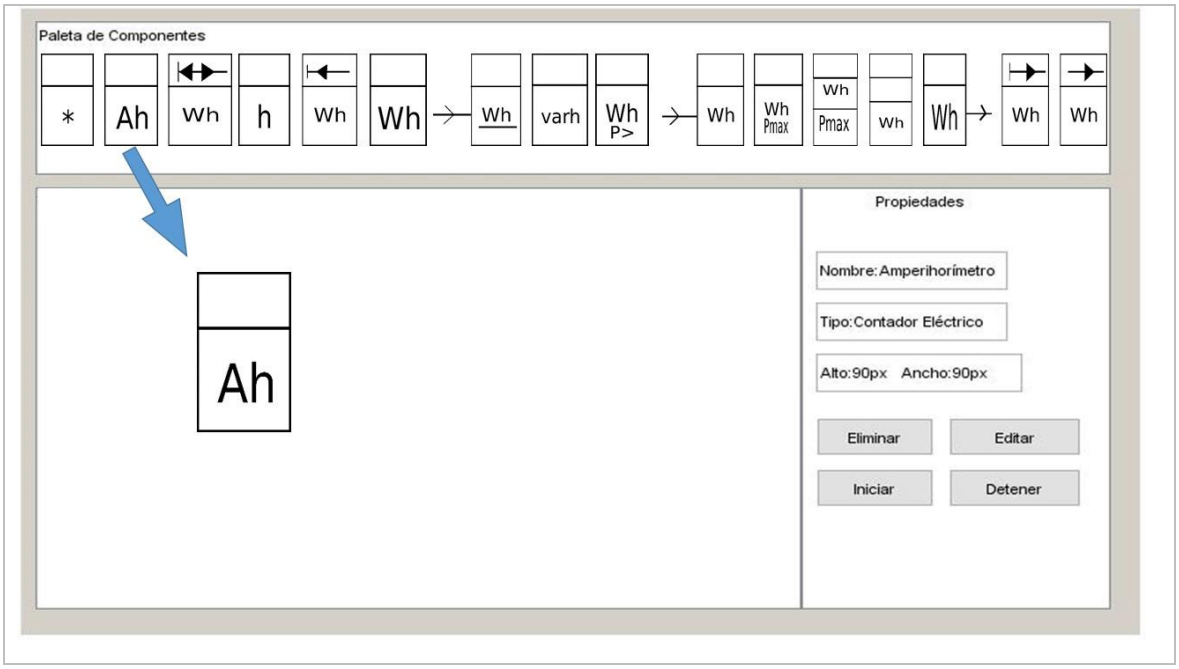


Tabla 7 Historia de usuario #3

Historia de Usuario	
Número: 3	**Nombre del requisito:** Rotar el componente gráfico en el despliegue.
Programador: Gustavo González Díaz.	**Iteración Asignada: 1**.
Prioridad: Alta.	**Tiempo Estimado:**1 semana.
Riesgo en Desarrollo: Problemas eléctricos, problemas técnicos.	**Tiempo Real:** 1 semana.
Descripción: El operador podrá modificar el componente gráfico de tipo contador eléctrico en el despliegue del editor gráfico estableciendo un ángulo de rotación, definiendo donde localizar el centro de rotación. El centro de rotación se puede localizar en distintas posiciones, en la parte inferior del componente, en la parte superior, en el centro, a la derecha o a la izquierda.	
Prototipo de interfaz:	

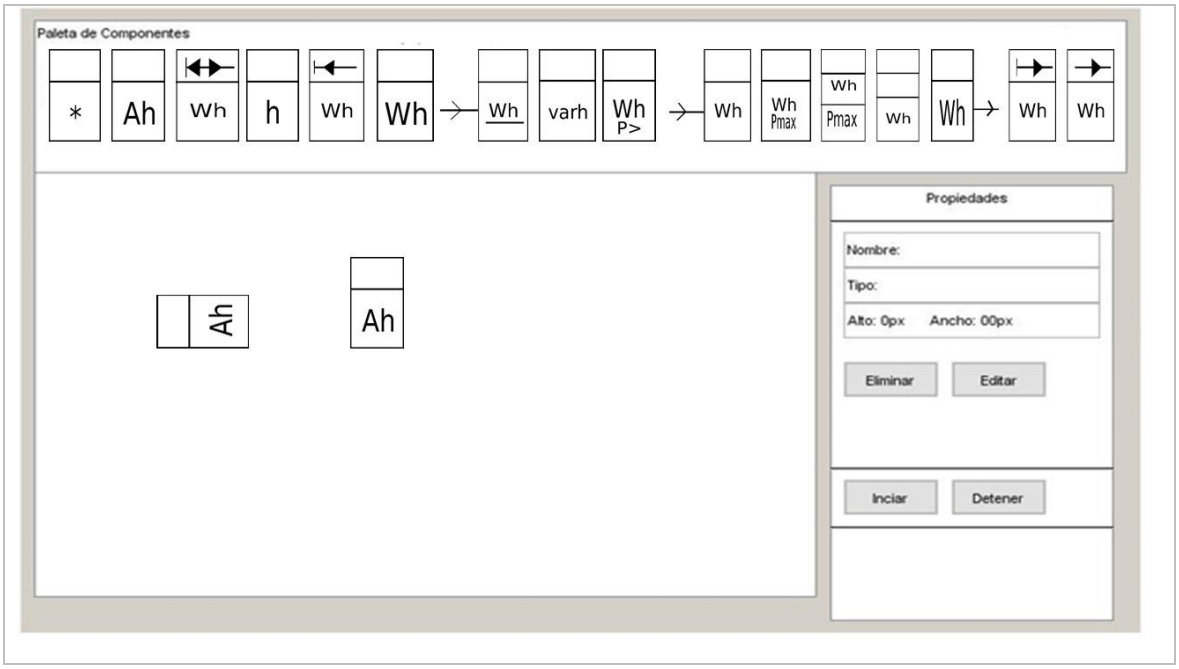


Tabla 8 Historia de usuario #4

Historia de Usuario	
Número: 4	**Nombre del requisito:** Redimensionar el componente gráfico en el despliegue.
Programador: Gustavo González Díaz.	**Iteración Asignada: 1**.
Prioridad: Alta.	**Tiempo Estimado:**1 semana.
Riesgo en Desarrollo: Problemas eléctricos, problemas técnicos.	**Tiempo Real:** 1 semana.
Descripción: El operador al realizar clic derecho encima del componente y seleccionar la opción propiedades, en la parte derecha del editor se muestra un inspector de propiedades, en donde el operador puede definirle el ancho o altura que desea que tenga el componente. También el operador al marcar el componente con un clic, situar el puntero en el borde izquierdo o derecho del componente y presionar el clic izquierdo, puede aumentar o disminuir el ancho o altura del componente a su preferencia.	
Prototipo de interfaz: **Prototipo de interfaz:**	

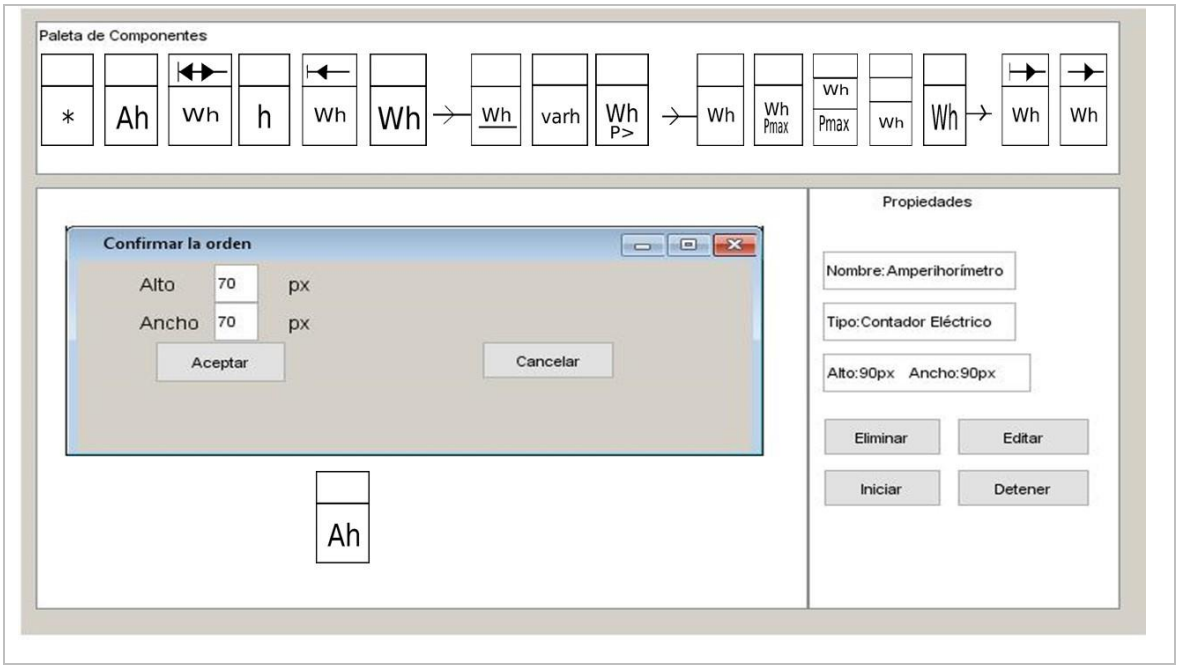


Tabla 9 Historia de usuario #5

Historia de Usuario	
Número: 5	**Nombre del requisito:** Mostrar propiedades del componente en el despliegue.
Programador: Gustavo González Díaz.	**Iteración Asignada: 1**.
Prioridad: Alta.	**Tiempo Estimado:**1 semana.
Riesgo en Desarrollo: Problemas eléctricos, problemas técnicos.	**Tiempo Real:** 1 semana.
Descripción: El operador podrá visualizar las propiedades del componente. A la derecha del editor se encuentra un inspector de propiedades el cual muestra las propiedades del componente	
Prototipo de interfaz:	

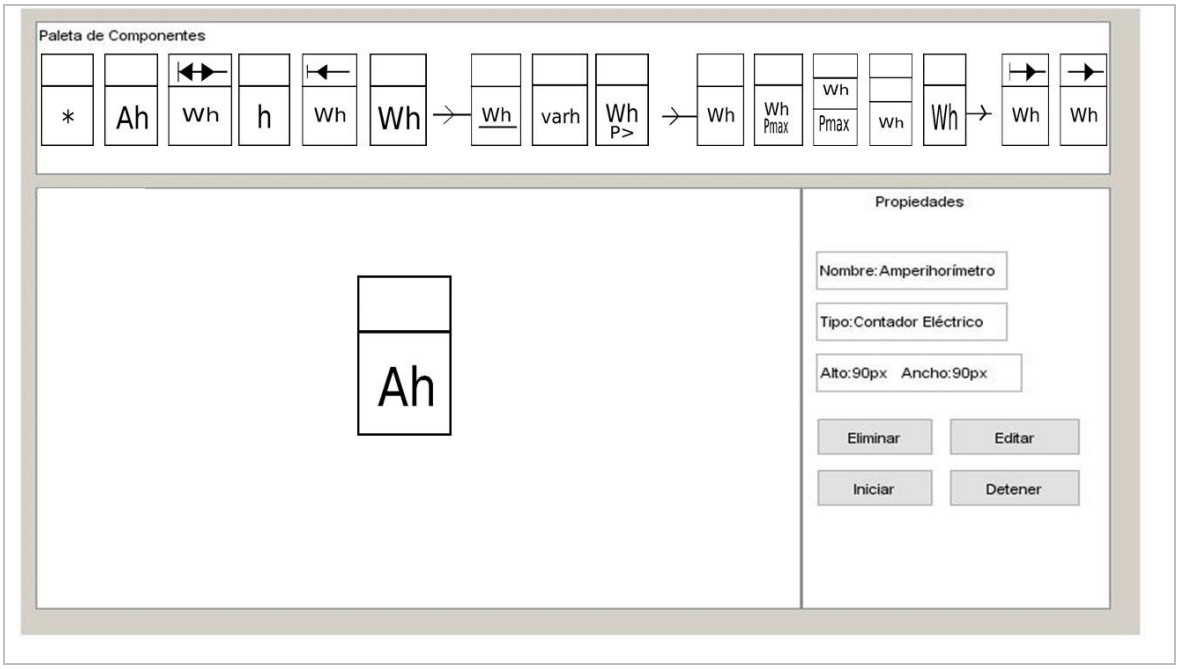


Tabla 10 Historia de usuario #6

Historia de Usuario	
Número: 6	**Nombre del requisito:** Actualizar las propiedades del componente gráfico.
Programador: Gustavo González Díaz.	**Iteración Asignada: 1**.
Prioridad: Alta.	**Tiempo Estimado:**1 semana.
Riesgo en Desarrollo: Problemas eléctricos, problemas técnicos.	**Tiempo Real:** 1 semana.
Descripción: El operador podrá configurar las propiedades del componente. A la derecha del editor se encuentra un inspector de propiedades el cual muestra las propiedades del componente al que se desee configurar.	
Prototipo de interfaz:	

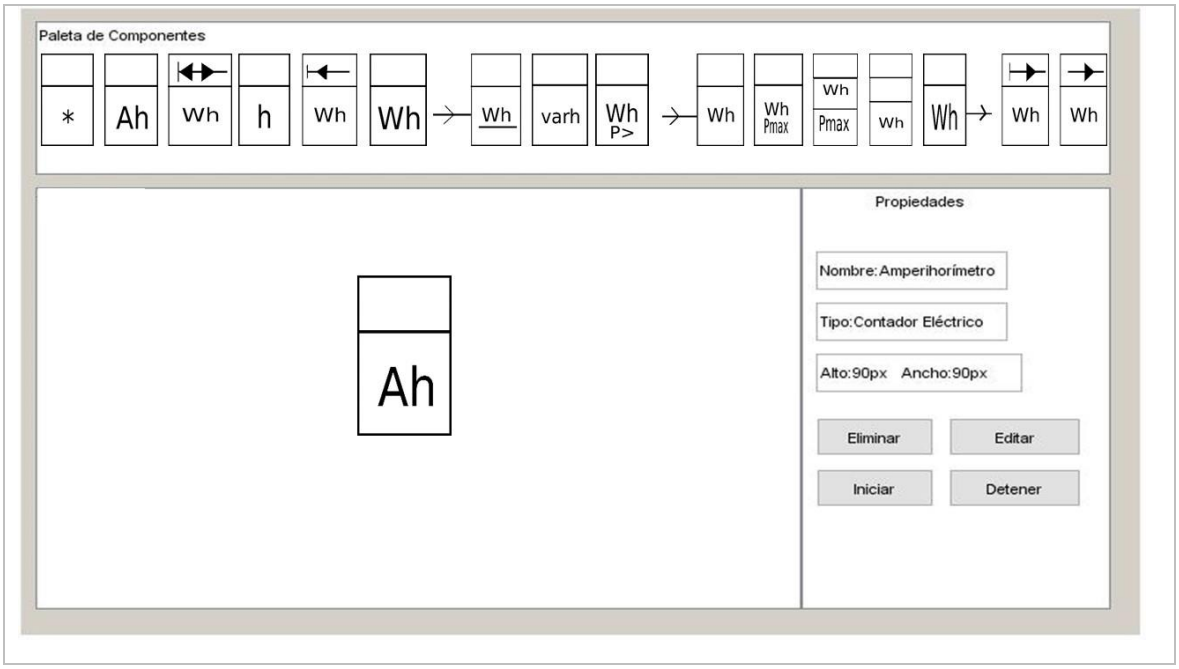


Tabla 11 Historia de usuario #7

Historia de Usuario	
Número: 7	**Nombre del requisito:** El sistema debe permitir mostrar el componente grafico en el visualizador.
Programador: Gustavo González Díaz.	**Iteración Asignada: 1**.
Prioridad: Alta.	**Tiempo Estimado:**1 semana.
Riesgo en Desarrollo: Problemas eléctricos, problemas técnicos.	**Tiempo Real:** 1 semana.
Descripción: El operador podrá visualizar el componente grafico en el visualizador.	
Prototipo de interfaz:	

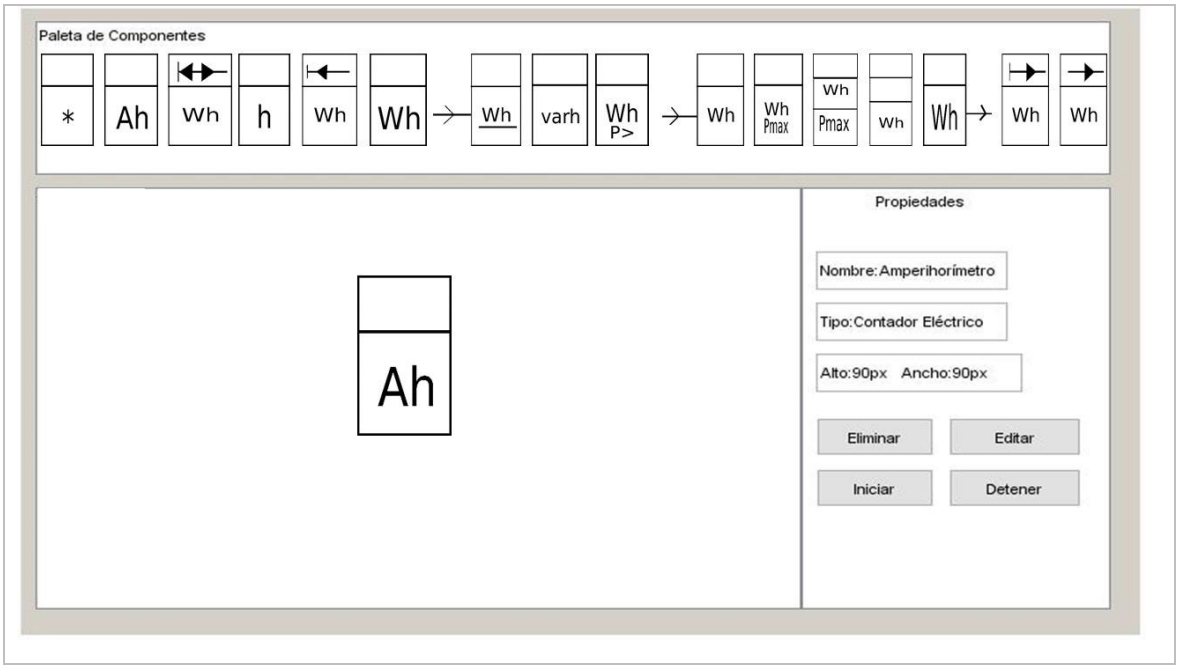


Tabla 12 Historia de usuario #8

Historia de Usuario	
Número: 8	**Nombre del requisito:** Eliminar el componente contador eléctrico del despliegue.
Programador: Gustavo González Díaz.	**Iteración Asignada: 1**.
Prioridad: Alta.	**Tiempo Estimado:**1 semana.
Riesgo en Desarrollo: Problemas eléctricos, problemas técnicos.	**Tiempo Real:** 1 semana.
Descripción: El sistema proporciona una interfaz al usuario donde el operador al dar clic izquierdo sobre el componente o al presionar la tecla *Delete* en el teclado, da la opción de eliminar el componente.	
Prototipo de interfaz:	

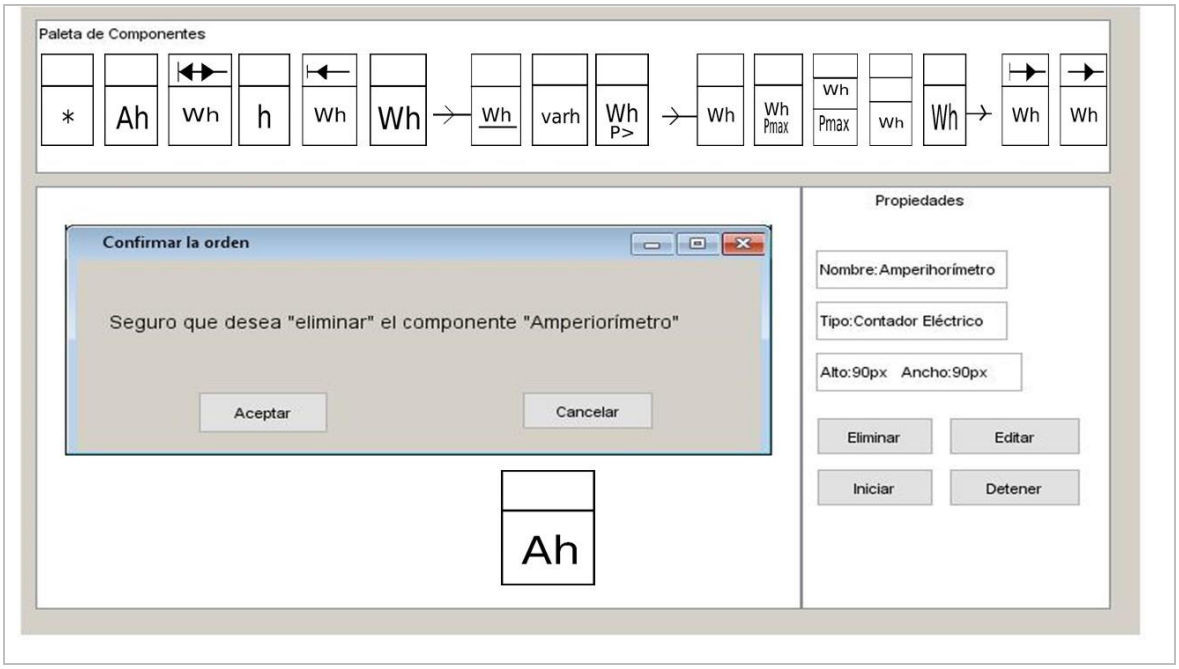
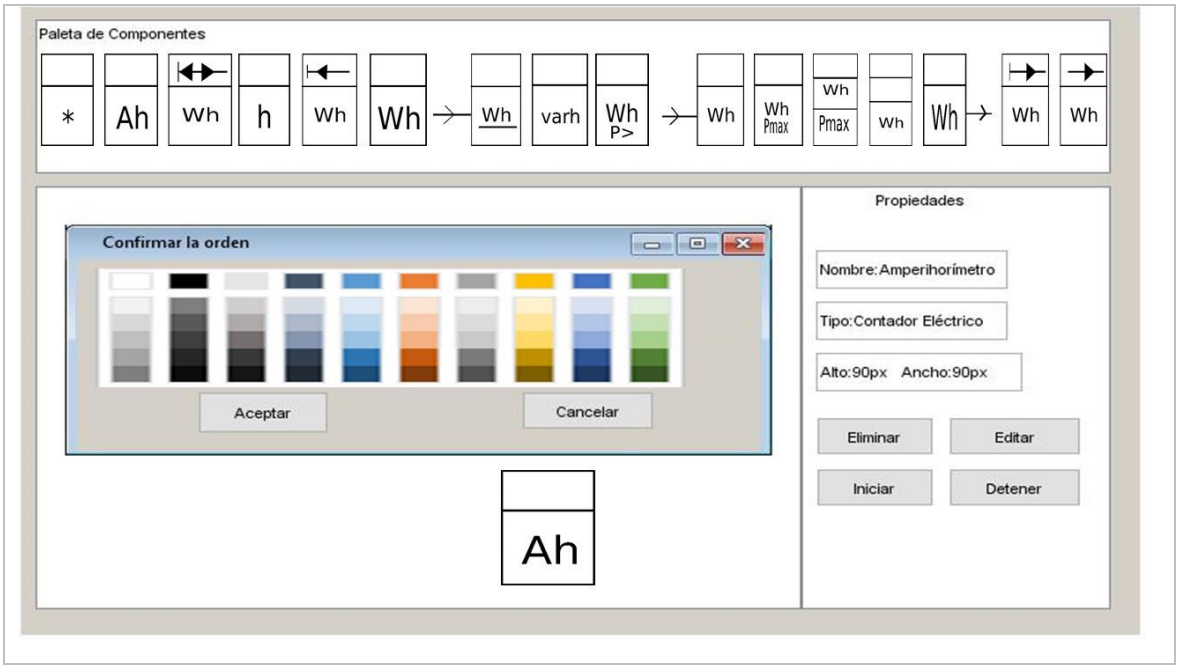


Tabla 13 Historia de usuario #9

Historia de Usuario	
Número: 9	**Nombre del requisito:** Definir color al componente eléctrico sobre el despliegue.
Programador: Gustavo González Díaz.	**Iteración Asignada: 1**.
Prioridad: Alta.	**Tiempo Estimado:**1 semana.
Riesgo en Desarrollo: Problemas eléctricos, problemas técnicos.	**Tiempo Real:** 1 semana.
Descripción: El operador podrá definir el color de las líneas del componente eléctrico sobre el despliegue.	
Prototipo de interfaz:	



2.4 Planificación del desarrollo.

Una vez definidas las historias de usuario (HU) se puntualiza la prioridad de cada una de ellas y la estimación de esfuerzo necesario para realizarlas para lo que se llevó a cabo un plan de 3 iteraciones las cuales son mostradas a continuación.

2.4.1 Estimación de esfuerzos e iteraciones por historia de usuario.

La siguiente tabla muestra estimación de esfuerzo para cada una de las historias de usuarios definidas en el desarrollo de la solución propuesta, el plan de iteraciones realizado y la duración total de las mismas.

Tabla 14 Estimación de esfuerzos

Historias de Usuario	Puntos de Estimación	Iteración	Duración total de las iteraciones (semanas)

Visualizar el componente en la paleta de componente del HMI.	0.9 semanas	1	3
Arrastrar el componente hacia el despliegue.	0,6 semanas		
Rotar el componente gráfico en el despliegue.	0,5 semanas		
Redimensionar el componente gráfico en el despliegue.	0,8 semanas		
Mostrar propiedades del componente en el despliegue.	1 semanas	2	1
Actualizar las propiedades del componente gráfico.	0,7	3	3
Mostrar el componente gráfico en el visualizador.	0,9		
Eliminar el componente contador eléctrico del despliegue.	0,5		

2.4.2 Plan de desarrollo

Luego de definidas las Historias de Usuarios (HU) y estimado el esfuerzo propuesto para su realización, se realizará el sistema en tres iteraciones, las cuales se describen detalladamente a continuación:

Iteración 1: La iteración tiene como objetivo realizar las HU 1, 2, 3 y 4 las cuales se encargarán de la configuración de los componentes correspondientes y garantizarán la representación gráfica de los componentes seleccionados.

Iteración 2: La iteración tiene como objetivo realizar la HU 5 siendo la encargada de mostrar las propiedades de cada uno de los componentes de tipo contador eléctrico.

Iteración 3: La iteración tiene como objetivo realizar las HU 6, 7 y 8 las cuales son las encargadas de mostrar el componente en el visualizador y las propiedades de los mismos para que sea configurado a conveniencia del cliente.

2.5 Diagrama de paquetes.

Este diagrama es el encargado de representar las dependencias entre los paquetes que componen un modelo. Es decir, muestra cómo un sistema está dividido en agrupaciones lógicas y las dependencias entre ellas (27).

En la figura 3 se muestra el diagrama de paquetes que conforma la propuesta de solución, representándose los paquetes modificados (Graphic, Graphics) y adicionados (ElectricMeter, Model, View):

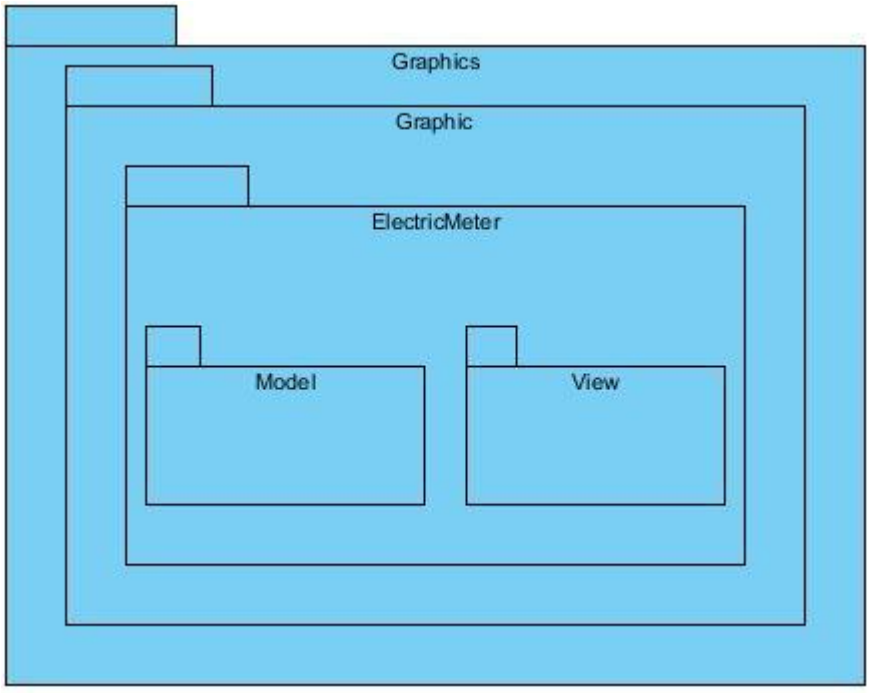


Figura 3: "Diagrama de paquetes "

El diagrama de paquetes está conformado por Graphics el cual tiene la responsabilidad de agrupar las entidades de la biblioteca de componentes gráficos. Dentro del paquete Graphic se almacenan los componentes: figuras básicas, válvulas, tanques, entre otros. Al paquete Graphic se le añade el paquete ElectricMeter el cual tiene la responsabilidad de agrupar el modelo y la vista de los componentes en los paquetes Model y View.

2.6 Diseño de Clases

El modelo de diseño permite producir varios modelos del sistema o producto que se va a construir, dicho modelo conforma el plan para la solución que será generada. Los diagramas de clases representan un conjunto de interfaces, colaboraciones y sus relaciones. Gráficamente son una colección de nodos y arcos. A continuación, se muestra el diagrama de clases de la solución propuesta con algunos de los componentes gráficos.

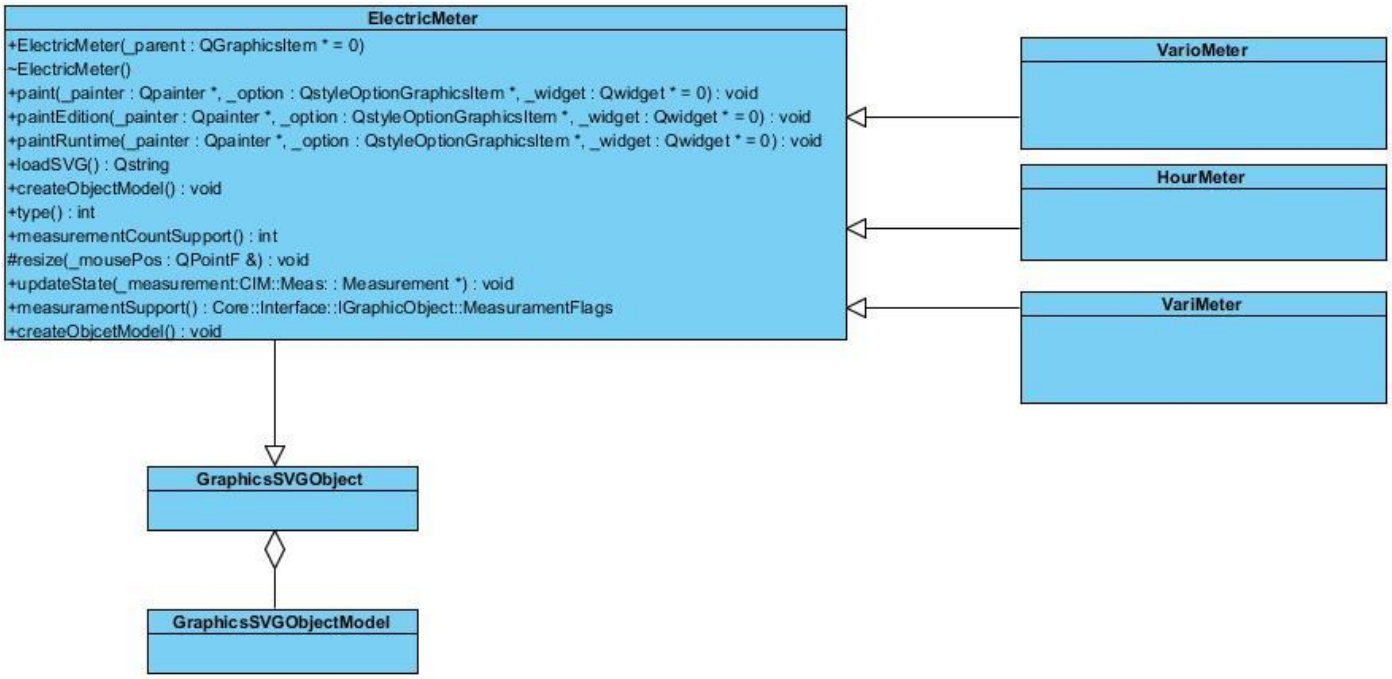


Figura 4: "Diseño de clases"

El diagrama de clases anteriormente expuesto está compuesto por el modelo: lo componen las clases GraphicsSVGObjectModel y GraphicsSVGObject, entre las cuales existe una relación de asociación. Por otra parte, la vista se compone por algunos de los componentes VarioMeter, HourMeter y VariMeter, y estas a su vez heredan de la clase padre ElectricMeter que es la clase controladora.

2.8 Patrones de Diseño.

Los patrones de diseño son un conjunto de prácticas de óptimo diseño que se utilizan para abordar problemas en la programación orientada a objetos. Un patrón de diseño proporciona un esquema para refinar sus subsistemas o componentes, o las relaciones entre ellos. Describe la estructura de la solución de un problema que aparece repetidamente; de componentes que se comunican entre ellos (28).

2.8.1 Patrones GRASP

Los patrones GRASP (General Responsibility Assignment Software Patterns, por sus siglas en inglés) o Patrones Generales de Software para Asignación de Responsabilidades, indican cual es la manera de asignar responsabilidades a objetos software (29).

Entre los patrones GRASP utilizados en la definición del sistema se encuentran:

- **Creador:** Permite asignar el responsable de la creación de una nueva instancia de alguna clase. Explica que clase es la encargada de crear objetos, en determinados escenarios de ejecución y guía la asignación de responsabilidades relacionadas con la creación de objetos. El propósito general de este patrón es encontrar un creador que se debe conectar con el objeto producido y se ve reflejado en la clase ElectricMeter().
- **Experto:** Asignar una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad. De forma general en el diseño del sistema se basa en asignar a cada clase la responsabilidad que solo ellas pueden realizar. Por lo que este patrón se encuentra en la clase ElectricMeter() y GraphicsSVGObjectModel().
- **Bajo acoplamiento:** ¿Cómo dar soporte a las bajas dependencias y al incremento de la reutilización? Asigne responsabilidades de manera que el acoplamiento (innecesario) se mantenga bajo.
- **Alta cohesión:** Facilita la solución al problema ¿cómo mantener manejable la complejidad?, mediante la asignación de responsabilidades de manera que la información almacenada en una clase sea coherente y esté relacionada con la clase. El uso de este patrón se ve reflejado en todo el diagrama debido a que cada clase almacena la información de ella misma de manera coherente.

2.8.2 Patrones GOF

Los patrones *Gang-of-Four* (“pandilla de los cuatro”) o comúnmente llamados Patrones GOF, descritos en el libro *Design Patterns* (30).

Según el libro GOF existen 3 tipos de patrones:

- De Creación: abstraen el proceso de creación de instancias.
- Estructurales: se ocupan de cómo clases y objetos son utilizados para componer estructuras de mayor tamaño.
- De Comportamiento: atañen a los algoritmos y a la asignación de responsabilidades entre objetos.

Para la construcción de la solución, con el objetivo de garantizar una buena práctica de programación, se emplearon los patrones GOF de comportamiento: observador y el método plantilla. A continuación, se presenta una breve descripción de cómo se definen estos patrones y su comportamiento.

- Método Plantilla: Este sencillo patrón resulta útil en casos en los que podamos implementar en una clase abstracta el código común que será usado por las clases que heredan de ella, permitiéndoles que implementen el comportamiento que varía mediante la reescritura (total o parcial) de determinados métodos. Este patrón se ve reflejado en los métodos paintRuntime (), paintEdition () y loadSVG ().

2.9 Conclusiones Parciales.

- Las herramientas seleccionadas facilitaron el proceso de diseño de la aplicación y la definición de sus funcionalidades y características.
- Entre los artefactos generados por la metodología usada se encuentra el diseño del modelo dominio el cual posibilitó el entendimiento del contexto de la solución.
- Las descripciones de las historias de usuario permitieron una mayor comprensión de los requisitos funcionales del sistema.
- La confección del diagrama de clases permitió describir a un nivel más detallado el desarrollo de la propuesta de solución.
- Se hizo uso de los patrones de diseño que dieron solución a problemas típicos y recurrentes que se encuentran en el desarrollo de una aplicación, lo que aportó la asignación de responsabilidades para las clases usadas en la implementación.

Capítulo 3: Implementación y Pruebas.

3.1 Introducción

En el presente capítulo se describirá las disciplinas de implementación y prueba del sistema. Se expondrá el estándar de codificación utilizado para el desarrollo de la solución, así como su respectivo diagrama de componentes y de despliegue. Se realizarán las pruebas correspondientes para la validación del sistema como las pruebas internas y de aceptación, utilizando el método caja negra.

3.2 Modelo de implementación

El diagrama de componentes es otro de los artefactos importantes que incluye el modelo de implementación. El mismo muestra elementos del modelo, tales como, los componentes y sus relaciones. Se utiliza para modelar la vista estática del sistema y muestra la organización y las dependencias lógicas entre los componentes de software, sean estos de código fuente, binarios o ejecutables.(31)

3.3 Diagrama de componentes

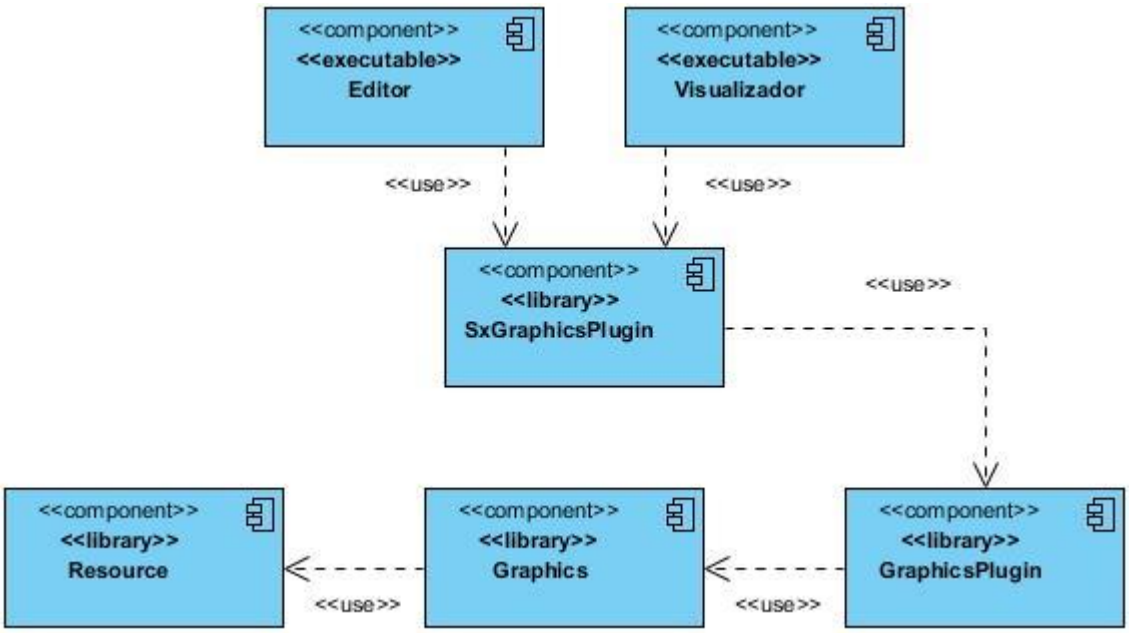


Figura 5: "Diagrama de componentes"

El diagrama de componentes antes dibujado está estructurado por ejecutables y bibliotecas:

- **Editor:** Es la aplicación donde se trabaja en el entorno de configuración para representar

los procesos del campo.

- **Visualizador:** Es una aplicación donde se supervisa el área configurada para interactuar con los componentes gráficos.
- **Biblioteca Graphics Plugins:** En esta biblioteca se almacenan los Plugins de los objetos gráficos.
- **Biblioteca Graphics:** Aquí están almacenados objetos gráficos que se utilizan para representar lo componentes gráficos, ya sea de forma simple o componentes complejos.
- **SxGraphics Plugins:** Es una interfaz de la biblioteca Graphics Plugins específica para los componentes de SAINUX2.0.
- **Resource:** Biblioteca donde se encuentran agrupadas cada una de las entidades encargadas de brindar las imágenes, los iconos y los diseños de los componentes en SVG.

3.4 Diagrama de Despliegue

Un diagrama de despliegue, es un diagrama que muestra la configuración de los nodos que participan en la ejecución y de los componentes que residen en ellos. Además, muestra, las relaciones físicas entre los componentes hardware y software en el sistema final, es decir, la configuración de los elementos de procesamiento en tiempo de ejecución y los componentes software (procesos y objetos que se ejecutan en ellos). Estarán formados por instancias de los componentes software que representan manifestaciones del código en tiempo de ejecución (32). A continuación, se muestra el diagrama de despliegue del sistema.

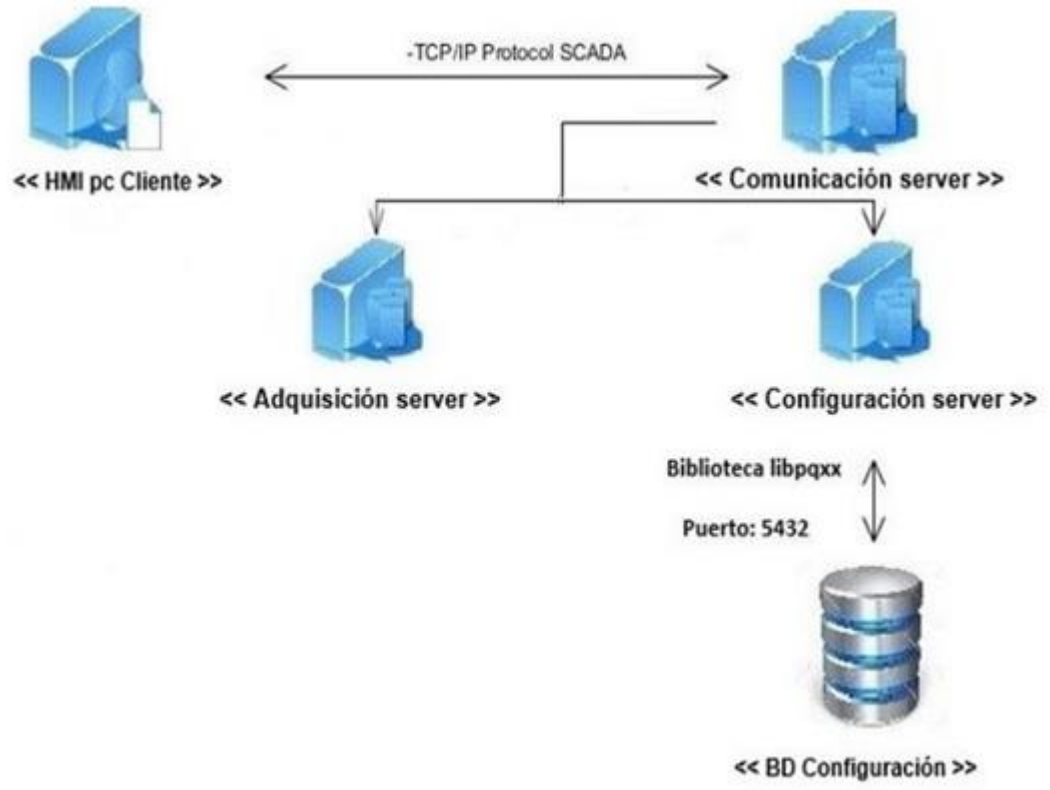


Figura 6: "Diagrama de Despliegue"

- **PC HMI:** En esta Pc se instala el módulo HMI, donde se encuentran desplegados el Ambiente Configuración y el Ambiente Edición, o solo uno de ellos.
- **Servidor Adquisición y Servidor Configuración:** Estos módulos se ejecutan en distintas Pc que pueden servir como servidores poniendo en evidencia la arquitectura distribuida que presenta el SCADA SAINUX.
- **BD Configuración:** El módulo Configuración se encuentran instalados con sus respectivas Bases de Datos conectados a través de los puertos 5432, utilizando la biblioteca libpqxx.
- **TCP/IP**: Es la conexión entre estos módulos se realiza utilizando comunicación con el protocolo SCADA desarrollado por el centro.

3.5 Estándar de Codificación

Los estándares de codificación, también llamados estilos de programación o convenciones de código, son convenios para escribir código fuente en ciertos lenguajes de programación. Estos estándares facilitan el mantenimiento del código, sirven como punto de referencia para los programadores, mantienen un estilo de programación y ayudan a mejorar el proceso de codificación, haciéndolo, entre otras cosas, haciéndolo más eficiente.(33)

Como la solución propuesta en este trabajo es parte del sistema SCADA SAINUX el estándar de codificación utilizado fue definido por el proyecto:

- Es importante especificar el nombre del autor, para ello se utilizan los comandos @autor.
- El código será escrito en inglés y la documentación en español.
- Las variables y funciones comienzan con letra minúscula. Cada palabra consecutiva en el nombre comienza con letra mayúscula.
  Ejemplo: void createObjectModel ();
- Los atributos de las clases deben empezar con m_ seguido del nombre del atributo, en el caso de atributos compuestos, la inicial de la segunda palabra debe comenzar con mayúscula.
  Ejemplo: m_posScreen
- Las funciones utilizan la nomenclatura camello.
- Los parámetros que recibe una función debe empezar con _ (guion bajo).
  Ejemplo: void paintRuntime (QPainter *_painter, const QStyleOptionGraphicsItem *_option, QWidget *_widget);
- Todas las funcionalidades y atributos deben seguir el siguiente formato: para documentar @brief Nombre del método.
- Ninguna función debe tener más de 200 líneas.
- Los valores de los numerativos deben ser con letras mayúsculas.
- Las secciones public, protected y private son declaradas en el orden expuesto.

3.6 Solución del Problema

A continuación, se muestra la solución del problema a resolver:

La biblioteca de componentes gráficos del HMI SAINUX 2.0 no le provee al mantenedor componentes para la representación de contadores eléctricos. Para solventar dicha situación el mantenedor utiliza o combina varias figuras geométricas simples como: polilínea, curvas, rombos, semicírculos y líneas. En la siguiente figura se muestran los accesorios que se utilizaban anteriormente.

Figura 7: "Componentes Básicos"

En la siguiente imagen vemos los componentes para la representación de los Contadores Eléctricos, al añadir un total de 16 componentes teniendo estas nuevas funcionalidades, vemos cómo se da cumplimiento a los problemas planteados sobre la representación de contadores eléctricos en el sistema.

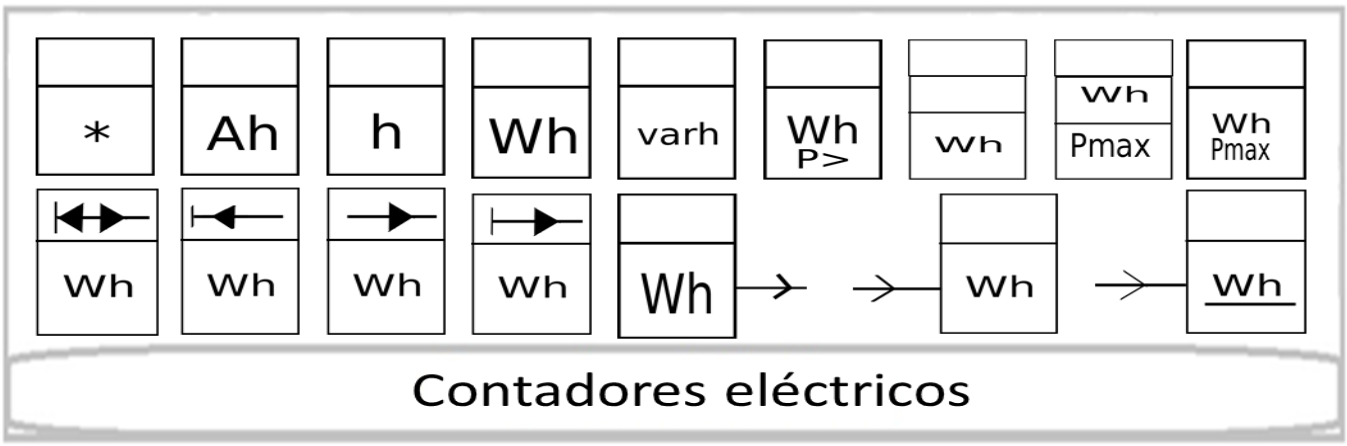


Figura 8: "Componentes de Contadores Eléctricos"

3.7 Pruebas

Las pruebas de software son un elemento crítico para la garantía de la calidad del software y representan una revisión final de las especificaciones, del diseño y de la codificación. El objetivo fundamental de las pruebas es descubrir diferentes clases de errores con la menor cantidad de tiempo y de esfuerzo. Aunque

las pruebas no pueden asegurar la ausencia de defectos; sí pueden demostrar que existen defectos en el software.(25)

Estas actividades se planean con anticipación y se realizan de manera sistemática. Cuando se aplican pruebas a un software es necesario tener en cuenta el objetivo que se persigue, debido a que las pruebas son agrupadas por niveles que se encuentran en distintas etapas del proceso de desarrollo. Como tal el objetivo es garantizar la calidad del producto desarrollado, además esta etapa implica:

- Verificar la interacción de componentes.
- Verificar la integración adecuada de los componentes.
- Verificar que todos los requisitos se han implementado correctamente.
- Identificar y asegurar que los defectos encontrados se han corregido antes de entregar el software al cliente.
- Diseñar pruebas que sistemáticamente saquen a la luz diferentes clases de errores, haciéndolo con la menor cantidad de tiempo y esfuerzo.

3.7.1 Método de caja negra

Para comprobar el correcto funcionamiento de las interfaces del software se aplicó el método de caja negra empleando la técnica de partición de equivalencia, la cual permitió examinar los valores válidos y no válidos de las entradas al sistema.

Son las pruebas que se llevan a cabo sobre la interfaz del software. Los casos de prueba pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce un resultado correcto. Estas pruebas permiten encontrar:

- Funciones que estén incorrectas o ausentes.
- Errores de interfaz.
- Errores en estructuras de datos o en accesos a las bases de datos externas.
- Errores de rendimiento.
- Errores de inicialización y terminación.

3.7.2 Aplicación de la prueba de caja negra

Los casos de prueba que se presentan a continuación pretenden demostrar que las funciones del sistema son operativas, que la entrada de los valores válidos y no válidos se acepta de forma adecuada y que se produce un resultado correcto.

Tabla 15 Diseño de casos de prueba

Descripción general				
Mostrar e interactuar con los componentes de tipo contador eléctrico en la paleta de componentes y el despliegue				
SC1 Mostrar el componente gráficos de tipo contador eléctrico en la paleta de componentes del HMI				
Escenario	**Descripción**	**Respuesta del sistema**	**Flujo Central**	**Resultados de la prueba**
EC1.1 Mostrar el componente en la paleta	1- Siguiendo la ruta del flujo central, en la pestaña de diseño, paleta de componentes del editor gráfico, se visualizan el contador eléctrico.	2- El sistema muestra el componente en la paleta de componentes.	Siguiendo la siguiente ruta: - En la paleta de componentes situada encima de la interfaz principal, se muestran en la región derecha los componentes gráficos tipo contador eléctrico.	Correcto con respecto a la respuesta del sistema.
SC2 Rotar el componente gráfico de tipo contador eléctrico en el despliegue.				
Escenario	**Descripción**	**Respuesta del sistema**	**Flujo Central**	**Resultados de la prueba**
EC2.1 Rotar el componente.	1- Siguiendo la ruta del flujo central, se selecciona el	2- Aparecerá un icono de rotación. 4-Se gira el componente.	Siguiendo la siguiente ruta: - En la región central de la	Correcto con respecto a la respuesta del sistema.

	componente haciendo clic en él. 3- Se gira el componente manteniendo el clic presionado y moviéndolo hacia la izquierda o hacia la derecha.		interfaz principal, se muestra el despliegue. - Presionando clic primario se muestran el icono de rotación.	
SC3 Arrastrar el componente gráfico de tipo contador eléctrico hacia el despliegue.				
Escenario	**Descripción**	**Respuesta del sistema**	**Flujo Central**	**Resultados de la prueba**
EC3.1 Mostrar componente	1- Siguiendo la ruta del flujo central, se muestra el componente arrastrado al despliegue.	2- Se muestra el componente tipo contador eléctrico en el despliegue.	Siguiendo la siguiente ruta: - En la región central de la interfaz principal, se muestra el despliegue.	Correcto con respecto a la respuesta del sistema.
SC4 Redimensionar el componente gráfico de tipo contador eléctrico en el despliegue.				
Escenario	**Descripción**	**Respuesta del sistema**	**Flujo Central**	**Resultados de la prueba**
EC4.1 Redimensionar componente	1-Siguiendo la ruta del flujo central, se selecciona el componente haciendo clic en él.	2-Aparecerán 6 puntos de edición en el contorno del componente.	Siguiendo la siguiente ruta: - En la región central de la interfaz principal,	Correcto con respecto a la respuesta del sistema.

	3-Se redimensiona el componente, haciendo clic en un punto y arrastrando, luego liberar.	4-Se modifica el tamaño del componente.	se muestra el despliegue. - Presionando clic primario se muestran los puntos de edición de tamaño.	
EC4.2 Redimensionar componente	1-Siguiendo la ruta del flujo central, se selecciona el componente haciendo clic en él, luego presionar clic secundario. 3- Se da clic en la opción “Propiedades”. 5- Se le asigna manualmente los valores de ancho y alto haciendo clic en el *textbox* al lado de “ancho” y “alto”.	2- Se despliega el menú de opciones. 4- Se muestra el Inspector de propiedades. 6- Se modifica el tamaño del componente.	Siguiendo la siguiente ruta: - En la región central de la interfaz principal, se muestra el despliegue. - Presionando clic secundario se despliega el menú de opciones del componente. - En la región derecha de la interfaz principal, se muestra el “Visor de propiedades”.	Correcto con respecto a la respuesta del sistema.
SC5 Mostrar todas las propiedades del componente gráfico de tipo contador eléctrico.				
Escenario	**Descripción**	**Respuesta del sistema**	**Flujo Central**	**Resultados de la prueba**

EC5.1 Mostrar propiedades	1-Siguiendo la ruta del flujo central, se selecciona el componente haciendo clic en él, luego presionar clic secundario. 3-Se selecciona la opción de “Propiedades”.	2-Se despliega el menú de opciones. 4-Se muestra el Inspector de propiedades.	Siguiendo la siguiente ruta: - En la región central de la interfaz principal, se muestra el despliegue. - Presionando clic secundario se despliega el menú de opciones del componente. - En la región derecha de la interfaz principal, se muestra el “Visor de propiedades”.	Correcto con respecto a la respuesta del sistema.
SC6 Actualizar las propiedades del componente gráfico de tipo contador eléctrico.				
Escenario	**Descripción**	**Respuesta del sistema**	**Flujo Central**	**Resultados de la prueba**
EC6.1 Configurar propiedades/Nombre	1-Siguiendo la ruta del flujo central, se selecciona el componente haciendo clic en él, luego presionar clic secundario.	2-Se despliega el menú de opciones. 4-Se muestra el Inspector de propiedades.	Siguiendo la siguiente ruta: - En la región central de la interfaz principal, se muestra el despliegue.	Correcto con respecto a la respuesta del sistema.

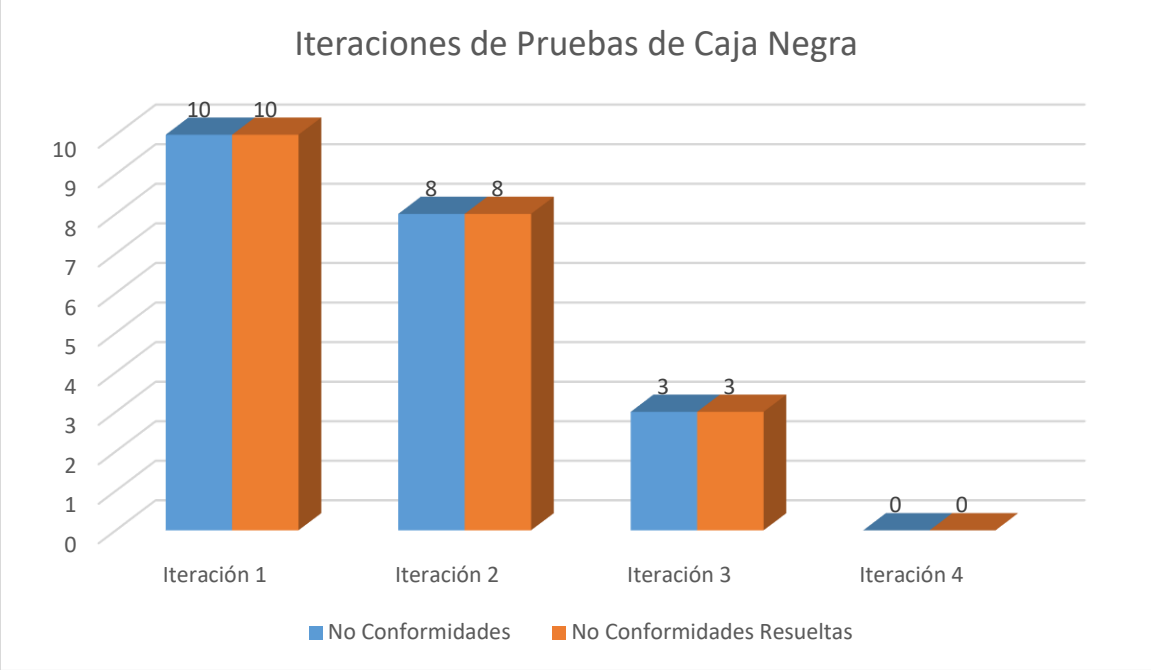
	3-Se da clic en la opción “Propiedades”. 5-Se da clic en “name”. 7-Escribir manualmente el nuevo nombre del componente.	6-Se muestra el nombre del componente seleccionado para editar. 8-Se visualiza el nuevo nombre al componente.	- Presionando clic secundario se despliega el menú de opciones del componente. - En la región derecha de la interfaz principal, se muestra el “Visor de propiedades”	
EC6.2 Configurar propiedades/Descripción	1-Siguiendo la ruta del flujo central, se selecciona el componente haciendo clic en él, luego presionar clic secundario. 3-Se da clic en la opción “Propiedades”. 5-Se da clic en “Descripción”. 7-Escribir manualmente la descripción del componente.	2-Se despliega el menú de opciones. 4-Se muestra el Inspector de propiedades. 6-Se muestra la descripción vacía del componente seleccionado para editar. 8-Se visualiza la descripción del componente.	Siguiendo la siguiente ruta: - En la región central de la interfaz principal, se muestra el despliegue. - Presionando clic secundario se despliega el menú de opciones del componente. - En la región derecha de la interfaz principal, se muestra el “Visor de propiedades”	Correcto con respecto a la respuesta del sistema.

			- En la región superior del visor de propiedades, segunda columna se muestra “Descripción”.	
SC7 Eliminar el componente contador eléctrico del despliegue.				
Escenario	**Descripción**	**Respuesta del sistema**	**Flujo Central**	**Resultados de la prueba**
EC7.1 Componente seleccionado en el despliegue.	1-Siguiendo la ruta del flujo central, se selecciona el componente haciendo clic en él y se presiona la tecla “*Supr”*.	2-Se elimina el componente correctamente del Runtime.	Siguiendo la siguiente ruta: - En la región central de la interfaz principal, se muestra el despliegue.	Correcto con respecto a la respuesta del sistema.
EC7.2 Componente seleccionado en el despliegue.	1-Siguiendo la ruta del flujo central, se selecciona el componente haciendo clic en él y se presiona clic secundario. 3-Se selecciona “Eliminar”. 5-Se selecciona “Aceptar”.	2-Se despliega un menú de opciones. 4-Aparece una ventana de confirmación de eliminado. 6-Se elimina el componente correctamente del Runtime.	Siguiendo la siguiente ruta: - En la región central de la interfaz principal, se muestra el despliegue. - Presionando clic secundario se despliega el menú de opciones del componente.	Correcto con respecto a la respuesta del sistema.

SC8 Cambiar color de las líneas al componente contador eléctrico sobre el despliegue.				
EC8.1 Configurar color de líneas	1. Se cambiará el color del componente seleccionado haciendo clic en él, luego presionar clic secundario. 3. Se da clic en la opción “Propiedades”. 4. Se da clic en “color de líneas”. 6. Se selecciona el color deseado dando clic en el mismo. 7. Se da clic en el botón “Aceptar”.	2. Se despliega el menú de opciones. 4. Se muestra el Inspector de propiedades. 5. Aparece paleta de colores disponibles. 8. Se muestra en el componente el nuevo color de líneas asignado.	Siguiendo la ruta: - Despliegue - Presionando clic secundario se despliega el menú de opciones del componente. - En la región derecha de la interfaz principal, se muestra el “Inspector de propiedades” - En la región central de la interfaz principal, se muestra la “Paleta de colores”	Correcto con respecto a la respuesta del sistema.
EC8.2 Configurar color de líneas	1. Se cambiará el color del componente seleccionado haciendo clic en él 2. Se da clic en “color de base”.	3. Aparece paleta de colores disponibles. 5. Se muestra en el componente el nuevo color de líneas asignado.	Siguiendo la ruta: - Despliegue - En la región derecha de la interfaz principal, se muestra el “Inspector de propiedades”	Correcto con respecto a la respuesta del sistema.

	4. Se selecciona el color deseado dando clic en el mismo. 7. Se da clic en el botón “Aceptar”.		- En la región central de la interfaz principal, se muestra la “Paleta de colores”	

Después de realizar las pruebas funcionales mediante el método de caja negra, se comprobó el correcto funcionamiento de la interfaz y la codificación del sistema. Cada problema detectado en el desarrollo del sistema fue resuelto a raíz del trabajo continuo del desarrollador, con un total de 10 no conformidades encontradas en la primera iteración, 8 en la segunda y 3 en la tercera, las cuales se dividieron en significativas y no significativas. A continuación, se representa lo expuesto anteriormente a través de la siguiente figura.



Iteraciones de Prueba de Caja Negra

3.8 Conclusiones Parciales

- Se ejecutaron las pruebas de caja negra las cuales validaron la implementación de los requisitos descritos en las historias de usuarios.
- Se presentó la distribución física del sistema y sus componentes mediante el diagrama de componentes, lo que permitió un mejor entendimiento de la distribución física y lógica del sistema.

Conclusiones generales

- La investigación inicial apoyada en la aplicación de los métodos científicos permitió identificar las funcionalidades y las tecnologías libres a utilizar en el desarrollo de la propuesta de solución lo que facilitó la nueva integración de componentes eléctricos al sistema SCADA SAINUX lográndose incrementar los grados de representación sobre el sector eléctrico en los procesos de visualización de la Interfaz Hombre Maquina.
- La utilización de la norma UNE-EN 60617 en la implementación de los componentes eléctricos del sistema SCADA SAINUX permite que el mismo esté en correspondencia con otras herramientas o sistemas orientados al sector eléctrico que utilicen dicha norma, lográndose el entendimiento de los componentes eléctricos.

Recomendaciones

Teniendo en cuenta los resultados obtenidos en esta investigación y basado en la experiencia adquirida, se recomienda:

- Identificar nuevos contadores que permitan ampliar los componentes eléctricos existentes.
- Incorporar funcionalidades de animación a los componentes eléctricos, para el caso en que lo necesitara el sistema.

Referencias Bibliográficas

1. PÉREZ LÓPEZ, Esteban. *Los sistemas SCADA en la automatización industrial*. 2015.

2. KARNOUSKOS. *Architecting the next generation of servicebased SCADA/DCS system of systems. IECON 2011 - 37th Annual Conference on IEEE Industrial Electronics Society.* 2011.

3. ANTUNEZ OJEDA, Yaima. *Plataforma de notificación de alarmas para sistemas SCADA a través de tecnologías de comunicaciones móviles.* 2014.

4. RODRÍGUEZ PENIN,. *Sistemas SCADA – Guía Práctica. España: MARCONBO.* 2007.

5. ARAGÓN CÁCERES. *Servidor de Comunicación con sistemas externos del SCADA - UX. Serie Científica de la Universidad de las Ciencias Informáticas*. 2011.

6. SÁNCHEZ OJEDA, Yosvani. *Componentes gráficos para la representación de signos de advertencia y señales de dirección en el SCADA SAINUX 2.0.* Universidad de las Ciencias Informáticas, 2017.

7. RIVERA ACOSTA., Keiger. *Desarrollo de la animación de movimiento compuesto en el SCADA SAINUX*. 2017.

8. About Citect SCADA 2018. [online]. 2018. Available from: https://www.citect.aveva.com/scada/citectscada/about-citect-scada-2018

9. AggreGate. [online]. 2018. Available from: http://aggregate.tibbo.com/

10. VAÑES BALLESTEROS. *Tipos de Contadores Eléctricos* [online]. 2014. Available from: https://germenstartup.wordpress.com/2014/05/17/tipos-de-contadores-electricos/

11. LÓPEZ DE PEÑALVER, Juan. Real Academia de Ingeniería. . 2012.

12. ROQUE DÍAZ, Pablo. Guía para determinar el consumo de energía eléctrica. [online]. 2011. Available from: http://www.cubasolar.cu/biblioteca/Energia/Energia33/HTML/articulo06.htm

13. *International Electrotechnical Commission* [online]. 2007. Available from: http://www.electropedia.org/iev/iev.nsf/display?openform&ievref=313-06-02

14. *Simbología Eléctrica Norma UNE-EN 60617 (IEC 60617)* [online]. 2014. Available from: http://www.infoplc.net/documentacion/236-esquemas-electricos/2161-

15. *Normas y Estándares Internacionales*. 2018.

16. Normas de instrumentacion industrial. [online]. 2018. Available from: https://es.slideshare.net/MarvinCampos2/normas-de-instrumentacin-industrial

17. Qué es SVG. [online]. 21 January 2017. Available from: http://aprendeweb.net/NT/svg/svg_1.php

18. FREEPRESS S, Mad. Inkscape: software libre para diseño vectorial. [online]. 2015. Available from: https://www.freepress.coop/inkscape-software-libre-para-diseno-vectorial/

19. *CORPORATION, NOKIA. Qt.* [online]. 2015. Available from: http://qt.nokia.com/products/developer-tools/.

20. VIÑOLO SOSA, Raydel Raúl. *Sistema Gestor de Proceso de Media v2* [online]. 2012. Available from: http://publicaciones.uci.cu/index.php/SC | seriecientifica@uci.cu.

21. RIVERA ACOSTA, Keiger. *Desarrollo de la animación de movimiento compuesto en el SCADA SAINUX*. 2017.

22. TALENS OLIAG, Sergio. *Curso de Programación en C++* [online]. 2015. Available from: http://www.uv.es/~sto/cursos/c++/curso95.pdf.

23. CMM.net. [online]. 2015. Available from: http://es.ccm.net/download/descargar-28127-visual-paradigm-for-uml-enterprise-edition.

24. Visión de Synergix de los Sistemas de Información y la Ingeniería del Software Tecnología y Synergix. [online]. 2013. Available from: https://synergix.wordpress.com/2008/07/10/modelo-de-dominio/

25. PRESSMAN, Roger S. *Ingeniería del Software. Un enfoque Práctico.* [online]. 3 February 2016. Available from: http://bibliodoc.uci.cu/pdf/8448111869.pdf.

26. PAULA IZAURRALDE, Maria. *Universidad Tecnológica Nacional Facultad Regional Córdoba Dirección de Posgrado Especialización en Ingeniería en Sistemas de Información* [online]. 2013. Available from: http://www.institucional.frc.utn.edu.ar/sistemas/lidicalso/pub/file/Tesis/Anteproyecto_Requerimientos_en_Metodolog%C3%ADas_Agiles.pdf

27. RAMOS CARDOZZO, Daniel. *Desarrollo de Sotware. Campus Academy*. 2014.

28. CCM. Patrones de Diseño. [online]. 2018. Available from: http://es.ccm.net/contents/224-patrones-de-diseno

29. MARCELLO VISCONTI, HERNÁN ASTUDILLO. Fundamentos de Ingeniería de Software. [online]. 2018. Available from: https://www.inf.utfsm.cl/~visconti/ili236/Documentos/08-Patrones.pdf.

30. GAMMA, E. Design Patterns. Elements of Reusable Object-Oriented Software. . 2018.

31. RIVERA ALVA, Eduardo. Arquitectura de software. Diagrama de componentes. [online]. 2016. Available from: http://es.scribd.com/doc/7884665/Arquitectura-de-Software-II-Diagrama-de-Componentes-yDespliegue

32. Diagrama de Despliegue ANALISIS Y DISEÑO DE SISTEMAS II. [online]. 2016. Available from: virtual.usalesiana.edu.bo/web/practica/archivo/despliegue.doc.

33. Estándares de Codificación .Net v.1.0.0.0. [online]. 2016. Available from: http: //serk.kualtus.com/codigo.htm